

DTIC FILE COPY

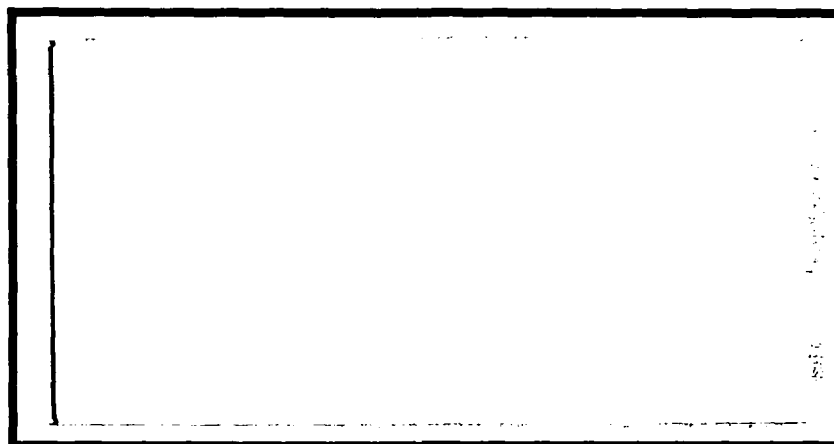
AD-A203 046



DTIC

JAN 18 1989

CH



DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

89

1 17 161

AFIT/GCS/ENG/88D-12

Redesign and Rehost of the BIG STICK
Nuclear Wargame Simulation

THESIS

Ruth M. Kalili
Captain, USAF

AFIT/GCS/ENG/88D-12

DTIC

UNCLASSIFIED

JAN 18 1989

H

Approved for public release; distribution unlimited

AFIT/GCS/ENG/88D-12

**Redesign and Rehost of the BIG STICK
Nuclear Wargame Simulation**

THESIS

**Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science (Information Systems)**

**Ruth M. Kalili, B.S., M.B.A.
Captain, USAF**

December, 1988

Approved for public release; distribution unlimited

Preface

The goal of this thesis project was to redesign and rehost the BIG STICK nuclear wargame simulation from the current mainframe-based system to the classroom microcomputers. The simulation is used at the Air Force Wargaming Center by the Air Command and Staff College as part of the nuclear warfare curriculum to teach intermediate level officers something about nuclear war planning.

The purpose of this thesis report is to present the background of the simulation; to describe the design of the microcomputer database, user interface, and simulation program of the new microcomputer-based BIG STICK simulation; and to describe the implementation of the new database and user interface.

Special thanks is extended to Capt Mark Roth, my thesis advisor, for his invaluable guidance and assistance with the development of this thesis effort. Thanks is also extended to my other committee members, Majors Bruce Morlan and James Howatt, for their willingness to help in this thesis effort.

I want to praise and thank my Lord and Savior, Jesus Christ, for His grace and power to help me overcome the seemingly impossible. Praise and thanks is also given to my friends and family for their prayerful support.

Finally, I wish to express a special aloha and mahalo to my son, Alan Reid, for the joy he brought into my life throughout my thesis and graduate degree endeavors.

Ruth M. Kalili



Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution	
Availability Codes	
Avail and/or	
Dist	Special
A-1	

Table of Contents

	Page
Preface	ii
Table of Contents	iii
List of Figures	vii
List of Tables	ix
Abstract	xi
I. Introduction	1
1.1 Background	1
1.1.1 Old BIG STICK Environment	2
1.1.2 New BIG STICK ENVIRONMENT	3
1.2 Problem Statement	5
1.3 Justification	5
1.4 Assumptions	5
1.5 Scope	6
1.6 Methodology	7
1.7 Sequence of Presentation	9
II. Requirements Analysis	10
2.1 Requirements Summary	10
2.1.1 Definition of the User	10
2.1.2 Customer Requirements	10
2.1.3 Validation of Customer Requirements	11
2.2 Design Considerations	12

	Page
2.2.1 Characteristics of the Student Operator	12
2.2.2 Characteristics of the Programmer-Maintenance Personnel	13
2.2.3 Software Limitations	13
2.2.4 Hardware Limitations	13
2.3 The Game	13
2.3.1 Force Selection	13
2.3.2 Force Deployment	14
2.3.3 Force Targeting	14
2.3.4 Force Employment	14
2.3.5 Example of the Game	15
III. Theoretical Development	19
3.1 Software Engineering Approaches	19
3.1.1 Life Cycle Model	19
3.1.2 Prototyping	19
3.1.3 Fourth Generation Techniques (4GT)	20
3.1.4 Hybrids	20
3.2 Database Design Approaches	20
3.3 User-Friendly Program Design Principles	21
3.3.1 System Security	23
3.3.2 User's Manuals	23
3.4 Summary	23
IV. System and Component Design	25
4.1 Database Design	25
4.1.1 Choice of Database Design Model	26
4.1.2 Development of the Relational Database Design	26
4.1.3 Database Exceptions	28

	Page
4.1.4 Partitioning of the Database	28
4.1.5 Reducing E-R Diagrams to Tables	31
4.1.6 Transforming ISA Links into Tables	31
4.1.7 Normalization	31
4.2 User Interface Design	31
4.2.1 Choice of Screen Design	32
4.2.2 Development of Screen Design	32
4.2.3 System Security	32
4.2.4 Other Interface Design Steps	34
4.3 Simulation Program Design	35
V. Implementation and Component Testing	37
5.1 Database Implementation	37
5.2 User Interface Implementation	38
5.2.1 Forms Development	38
5.2.2 Menu Options Modification	39
5.2.3 Frame Development	40
5.2.4 Interface Implementation Problems	40
5.2.5 Interface Component Testing	41
5.3 Maintenance Program Implementation	42
VI. Design Summary and Recommendations	43
6.1 Design Summary	43
6.2 Recommendations	44
Appendix A. New Deployment PD Form Numbering Scheme	45
Appendix B. Database Relations	46
Appendix C. IDEF ₀ Diagrams	61

	Page
Appendix D. Simulation Flowcharts	69
Bibliography	82
Vita	84

List of Figures

Figure	Page
1. BIG STICK Simulation Environment	4
2. Software Development Cycle	8
3. Simplified Flowchart of Game Play	16
4. Bomber Simulation Profile	18
5. BIG STICK E-R Diagram	27
6. Site Entities and Relationships	29
7. Force Entities and Relationships	30
8. Sample Prototype Screen Layout	33
9. A2-Simulation IDEF₀ Diagram	36
10. Final Force Selection Screen	39
11. A0 - BIG STICK Environment IDEF₀ Diagram	62
12. A25 - Force Employment IDEF₀ Diagram	63
13. A252 - Retarget IDEF₀ Diagram	64
14. A253 - Negotiate IDEF₀ Diagram	65
15. A254 - Gaming IDEF₀ Diagram	66
16. A2541 - Get Options IDEF₀ Diagram	67
17. A255 - Post_War Actions IDEF₀ Diagram	68
18. A251 - Defense Systems Pre_War Actions Flowchart	70
19. A2543/5 - SAM and SAM-D Enroute & Reports Flowchart	71
20. A2543/5 - Interceptor Enroute & Reports Flowchart	72
21. A251 & A2542 - Bomber Pre_War Actions & Launch Flowchart	73
22. A2543/4/5 - Bomber Enroute, Impact, & Reports Flowchart	74
23. A2543/4/5 (cont) - Bomber Enroute, Impact, & Reports Flowchart	75
24. A251 - Fighter Pre_War Actions Flowchart	76
25. A2542/3/4/5 - Fighter Launch, Enroute, Impact, & Reports Flowchart	77

Figure	Page
26. A251 - SNF Pre-War Actions Flowchart	78
27. A2542/3/4/5 - SNF Launch, Enroute, Impact, & Reports Flowchart	79
28. A251 - NNF Pre-War Actions Flowchart	80
29. A2542/3/4/5 - NNF Launch, Enroute, Impact, & Reports Flowchart	81

List of Tables

Table	Page
1. Blue Site Table Description	37
2. Force Summary Table Description	38
3. New PD Form Schedule	45
4. Blue Red Select Force Relation	46
5. Blue Red Adjacent Sector Relation	46
6. Blue Red Adjacent Area Relation	47
7. Blue Red ASW Operational Sector Relation	47
8. Submarine Schedule Relation	47
9. Bomber Type Relation	48
10. Blue Red Bomber Legal Relation	48
11. Blue/Red Bomber Load Relation	48
12. Fighter Type Relation	49
13. Blue Red Fighter Legal Relation	49
14. Strategic Nuclear Force (SNF) Type Relation	49
15. Blue/Red SNF Legal Relation	50
16. Non-Nuclear Force (NNF) Type Relation	50
17. Blue/Red NNF Legal Relation	50
18. Probability of Delay Relation	50
19. Probability of Damage Relation	51
20. Probability of Escape Relation	51
21. Game Constants Relation	51
22. Blue/Red Battery Relation	52
23. Blue/Red City Site Relation	52
24. Blue/Red War Site Relation	52
25. Blue/Red Economic Site Relation	52

Table	Page
26. Blue SCM /Red IRBM Site Relation	53
27. Blue/Red Bomber Site Relation	53
28. Blue/Red Fighter Site Relation	53
29. Blue/Red Auxiliary Air Field Relation	53
30. Blue/Red Port Site Relation	54
31. Escape Sector Relation	54
32. Blue /Red Sea Sector Relation	54
33. Blue /Red ICBM Site Relation	54
34. Blue /Red Activates Relation	55
35. Blue/Red IC3 Site Relation	55
36. Blue/Red ABM Site Relation	55
37. Blue/Red AD Site Relation	55
38. Blue/Red ADCC Site Relation	56
39. Blue/Red Deploy Force Relation	56
40. Blue/Red Deploys Relation	56
41. PD Form Schedule Relation	56
42. Blue/Red Force Relation	57
43. Blue/Red Target Force Relation	57
44. Blue/Red Units Relation	57
45. Blue/Red Mated With Relation	57
46. Blue/Red Stored On Relation	58
47. Blue/Red Bomber Vehicle Relation	58
48. Blue/Red Bomber Targets Relation	58
49. Blue/Red Fighter Vehicle Relation	59
50. Blue/Red Fighter Bombs Relation	59
51. Blue/Red SNF Vehicle Relation	59
52. Blue/Red SNF Warheads Relation	60
53. Blue/Red NNF Vehicle Relation	60

Abstract

The strategic nuclear wargame BIG STICK is a two-sided, interactive, computer simulation used by the Air Command and Staff College to assist students in learning about real-world nuclear war planning.

Currently, the simulation is played on the Honeywell H6000 mainframe. Shortcomings of the simulation are the "user-hostile" environment, a rigid input format, the unforgiving and inflexible user interface, and the fixed file system that makes data changes and program enhancements difficult.

The purpose of this thesis effort was to redesign and rehost the BIG STICK simulation to the Zenith Z-158 classroom microcomputers.

Game sites, fixed and controlled force assets, expected value probabilities, and exercise constraints are now stored in a relational database designed using the entity-relationship (E-R) model.

The user interface was redesigned using general user-friendly program principles and guidelines to provide a screen oriented environment for students to enter force selection, deployment, targeting, and employment inputs. Reports reflecting the results of student inputs were designed and developed as part of the interface using the PC INGRES database management system.

The actual game play portion of the simulation was designed using top-down, hierarchical decomposition. Implementation of the design into program code is not included in the scope of this thesis.

Redesign and Rehost of the BIG STICK Nuclear Wargame Simulation

1. Introduction

Software engineering, database design, and user-friendly program principles and techniques have been combined in this thesis project to create a microcomputer-based wargame simulation. This simulation is developed to support the Air Force Wargaming Center in their effort to enhance the wargaming capabilities of the professional military education schools.

1.1 Background

BIG STICK is used by the Air Command and Staff College (ACSC), a professional military education (PME) school for intermediate level officers, as part of the nuclear warfare curriculum. Students first enter the planning phase of the exercise which consists of force structuring and deployment and development of force operation strategies and objectives. The students then proceed to the simulation phase in which they test their strategies against a reactive opponent. Although all targets, costs, and weapon system capabilities are fictitious, the exercise affords PME students an opportunity to test and confirm concepts learned relating to national security policy, principles of war, current aerospace doctrine, resource management, and the operational planning process in the event of a nuclear war [1]. The exercise is designed to help students use skills developed in analytical techniques, staff problem solving, and leadership under stress.

The BIG STICK simulation involves the procurement of selected conventional and nuclear forces and defense systems, the deployment and targeting of these systems, and their ultimate employment. Intelligence and operation & maintenance budget considerations are also included in the simulation.

Forces are employed in the wargame which simulates the first 12-15 hours of a nuclear exchange. The war takes place in the late 1990s between the blue side consisting of the

U.S. and the Blue European forces and the red side consisting of the U.S.S.R. and its European satellite forces. Because of past Strategic Arms Reduction Talks (START), the nuclear capabilities of both sides are essentially equivalent. Beginning with a European tactical air battle along the blue-red common border, the war escalates to the use of non-strategic nuclear systems. "After this battle, each leg of the strategic TRIAD (ICBM, SLBM, Bomber) is exercised as the war expands to global proportions" [20:682-683].

The BIG STICK simulation is a two-sided, interactive, computer wargame [6]. The wargame simulates a nuclear war using a stochastic or probabilistic, event-driven model. Fixed expected value probabilities are used to determine the success or failure of simulated events. Some of these events are launch success, aircraft no abort, missile reliability, success of avoiding area and point defenses, target found, and target destroyed. Throughout the game, these expected value probabilities are compared to random numbers. If the random number generated is less than or equal to the expected value probability of a particular event, then that event is a success. This use of random numbers is known as the Monte Carlo method [8].

The newly created Air Force Wargaming Center (AFWC) maintains the exercise and has provided microcomputers to enhance the wargaming capabilities of the PME schools.

1.1.1 Old BIG STICK Environment. BIG STICK began as a game played strictly with pencil, paper, charts, and graphs [21]. Currently, the simulation is written in FORTRAN and runs on a Honeywell H6000 mainframe located at Gunter AFB and is connected using Hayes SMARTMODEM modems to the classroom terminals at Maxwell AFB, site of ACSC.

The planning and analysis portion which precedes the simulation portion of the BIG STICK exercise has already been implemented on the new microcomputers. In 1987, Major Charles Williams developed the BIG STICK Planning and Analysis Tool (BSPAT) written in the BASIC programming language to assist students in the planning and analysis of force selection, deployment, and targeting decisions [21]. The results are stored in a data file and hard copy printouts are presented to the team chairperson for evaluation and later are manually input to the mainframe system.

A shortcoming of the current simulation is the "user-hostile" environment in which the game is played. The program inputs must follow a rigid format and the user interface is unforgiving and inflexible because of the fixed file management system and the fixed reporting capabilities. Similar to the implementation of the Joint Planning (JPLAN) exercise, the implementation of BIG STICK is:

closely coupled to FORTRAN conventions, [that] the user is forced to enter the lines of data with commas and spaces placed exactly as required or he must start over when an input error occurs. In summary, the students spend too much time wrestling with the current simulation detracting from the learning objectives. [11:3]

Other limitations of the current mainframe-based simulation prevail and are primarily due to the storage method of the expected value probabilities, game assets and targets, and exercise restrictions. The fixed, hard-coded file system used to store the large amounts of data required by the simulation makes data changes and program enhancements difficult.

1.1.2 New BIG STICK Environment. Three of the four parts of the simulation portion of the BIG STICK exercise are implemented on the Zenith Z-158 microcomputer with 640-kilobyte random access memory (RAM) and 10-megabyte Bernoulli hard disk drive cartridges. Completion of the coding and testing phases of the final portion of the simulation will result in a fully-implemented exercise that may be played on any IBM/XT or IBM/AT-compatible microcomputer having similar memory and disk storage capabilities.

Six components create the BIG STICK microcomputer-based system environment: the simulation application program; the database which replaces the fixed, hard-coded file system; the two interfaces between the database and the application program; the report section; and the interconnection between the two student, "end user", teams. The relationships between these components are shown in Figure 1.

This microcomputer environment allows force selection, deployment, targeting, and employment inputs to be entered without the interference of any input constraints imposed under the old BIG STICK environment. Remaining restrictions are those prescribed by the rules of exercise.

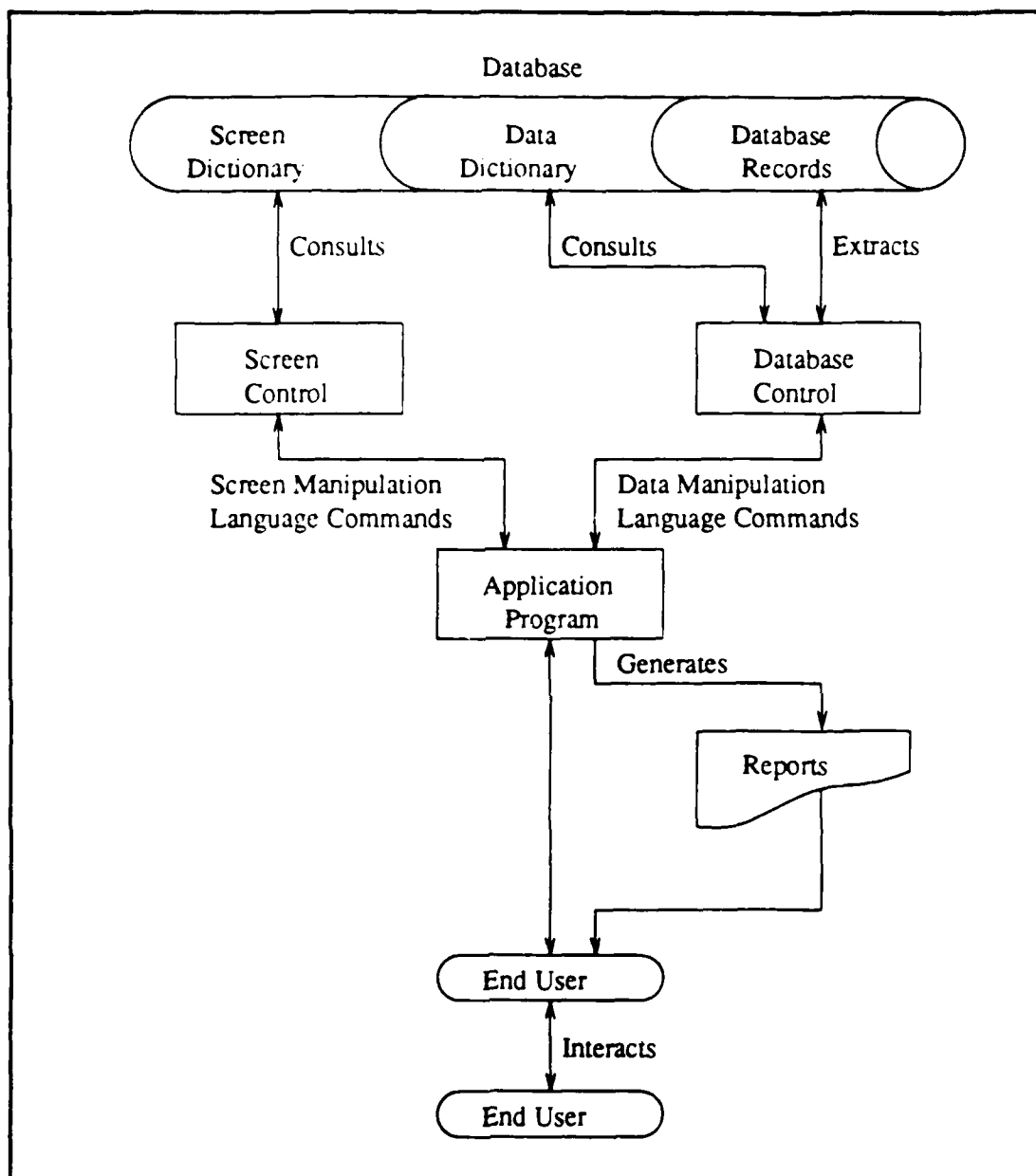


Figure 1. BIG STICK Simulation Environment

1.2 Problem Statement

The purpose of this thesis project is to combine software engineering, database design, and user-friendly program principles and techniques in an effort to redesign and rehost the simulation portion of the BIG STICK exercise from the current mainframe-based system to a microcomputer-based system.

1.3 Justification

Coupled with the BPSAT, the microcomputer implementation of the simulation portion of the BIG STICK exercise will create a fully-independent war exercise that can be planned and played on the ACSC classroom microcomputers [21]. The microcomputer-based system will also make possible the use of the wargame in non-resident seminar and correspondence PME programs, a long range goal of the PME schools and the Air Force Wargaming Center [6].

This thesis effort will produce a more user-friendly environment that will allow students to spend less time learning the computer syntax and more time playing the game. The more user-friendly environment will also free the faculty instructors from instructing students on the use of the computers, allowing them to emphasize the learning objectives of the exercise.

Maintenance and enhancement will be easier as a result of this effort, with the document supported database component.

1.4 Assumptions

The development of the BIG STICK exercise on the microcomputer-based system meets the learning objectives of the Air Force Wargaming Center (AFWC). One change to the exercise is incorporated in this development. At the request of Lt Colonel Francis Walker, AFWC BIG STICK exercise controller, the Damage Assessment Satellite System (DASS) reports which had been provided to students immediately following the end of every employment time period, are now provided after a delay of two time periods. One

other minor modification is the renumbering of the planning document (PD) forms used in deployment (see Appendix A).

The development does not include redesigning the BIG STICK Planning and Analysis Tool (BSPAT), but will allow student teams to transfer the output data directly into the simulation.

Future upgrades of the exercise are unknown, but the implementation will allow easy modification of the simulation to include new systems such as the strategic defense systems.

1.5 Scope

The scope of this thesis is limited to the six design components of the BIG STICK simulation presented earlier and other supporting units.

The relational database model will be used to design the BIG STICK database and Relational Technology's PC INGRES database management system will be used to store all screen and data dictionary entries along with the database records required by the simulation.

The user-friendly screen control system will be developed using the PC INGRES Forms tool and 4GL programming language to build the frames and control the movement between frames and applications. In INGRES, a *frame* consists of forms or the actual screen layouts and *menu options* located at either the top or bottom of the form. This menu options capability will be used to develop an on-line help system that will provide to the terminal display all the information normally contained in a user's manual. An *application* is used to describe a set of frames and is created using the PC INGRES Applications-By-Forms (ABF) package.

The database control system is already developed through other features of PC INGRES. PC INGRES uses the Structured Query Language (SQL) to manipulate the database. These SQL commands can be embedded in a host programming language such as Microsoft C provided there is a preprocessor available. Here a C preprocessor is required and is available with PC INGRES.

Reports will be generated by capabilities provided by the PC INGRES Report Writer package.

The simulation application program component, which represents the force employment phase of the simulation, will be designed using IDEF₀ and flowchart diagrams.

The connection between the microcomputers used by opposing student teams will be done through swapping of floppy disks since alternatives are not supported by the Wargaming Center's current hardware and software capabilities.

In addition to these six components, the scope of the thesis project includes a maintenance program which will allow maintenance programmer personnel to access the database tables for modification and enhancement of the exercise and project documentation. The software project will be documented with a user's manual and a maintenance manual.

1.6 Methodology

The overall system development methodology adopted in this thesis effort is the fourth generation techniques (4GT) paradigm described by Pressman [16]. The 4GT software development approach consists of four iterative phases: the requirements gathering phase, the design strategy phase, the implementation using a fourth generation language (4GL) phase, and the product phase. This approach is modified to separately identify a prototype phase and the modified version is presented in Figure 2.

This thesis project begins with a requirements gathering step. The user and the user requirements are defined. Requirements for all system components are also identified. A product of this phase was presented earlier (see Figure 1).

Storyboarding, or the building of a paper prototype, of the screens for the simulation is performed. Prototyping is used in this thesis project to identify, validate, and refine system, software requirements.

In this large of a software effort, design strategy is necessary in order to meet the four goals of software engineering which are: modifiability, efficiency, reliability, and understandability [4]. A part of the overall design strategy of the thesis project is the need to decompose the system into components and then design the components using prin-

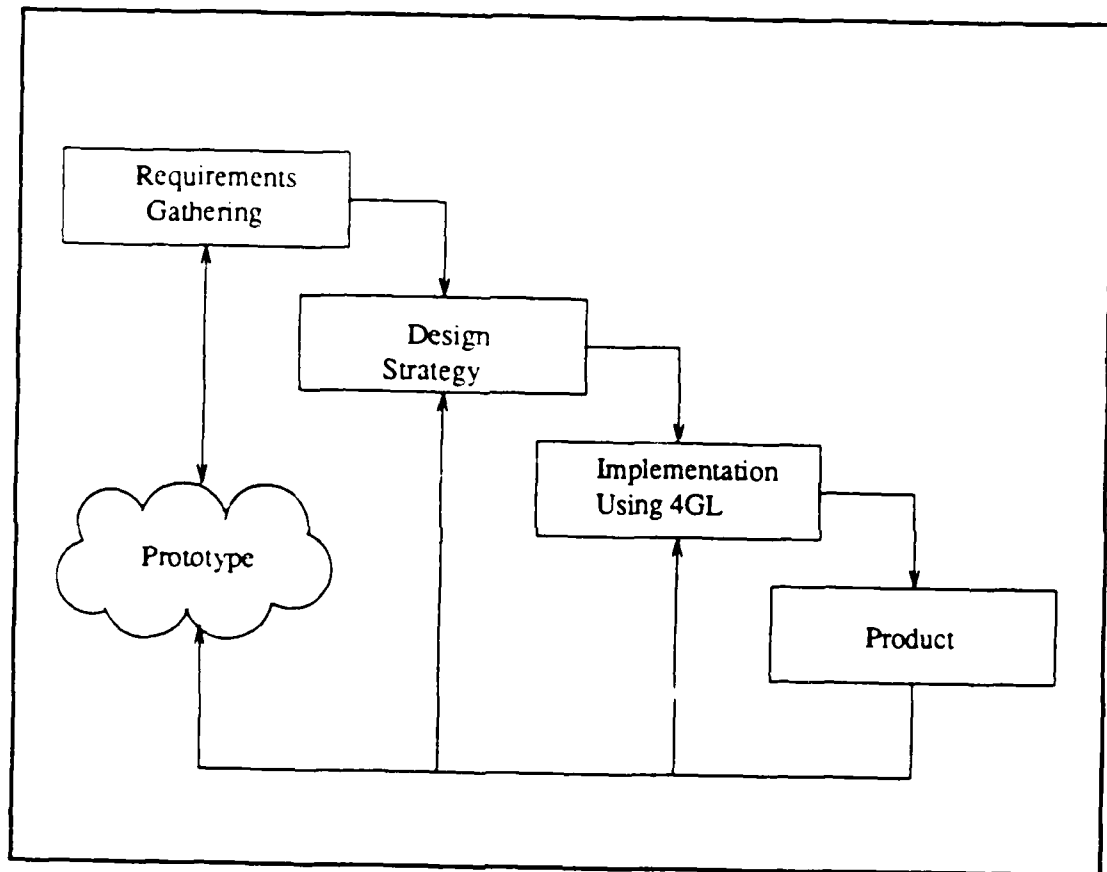


Figure 2. Software Development Cycle [16]

ciples and techniques unique to the specific component. For the design of the database component, database objects or entities and relations are identified and normalized using an entity-relationship (E-R) design diagram. The design of the screen or user interface component adheres to user-friendly program design techniques. These techniques are used to determine screen layout and use of color, brightness change, reverse video, and other features in the screen display. Report design is similar to screen layout design. The design of the application program or simulation code is developed through the use of IDEF₀ function diagrams (a product of SofTech, Inc.) and flowcharts.

Implementation of the database, user interface, and the maintenance program is accomplished incrementally using third and fourth generation languages. This portion is conducted using Boehm's spiral model of incremental development and integration of software [3].

In the product phase of the thesis project, component testing, partial system integration, and partial system testing is performed. Product descriptions including user and maintenance manuals developed continuously from the start of the project are prepared for final format.

1.7 Sequence of Presentation

This thesis captures the design process used and design decisions made in developing the wargame software. This thesis is intended to provide the reader the information or access to the information necessary to be able to duplicate, parallel or enhance the thesis project in developing a microcomputer-based wargame simulation.

In the next chapter a more complete, detailed expansion of the problem definition is provided. Chapter III reviews existing theory applicable to the problem. Then in Chapter IV the system component designs are developed. Here the focus is on the major decisions and considerations made in the development of each design. Following the description of the design development, discussion of the significant problems encountered in the implementation and test of each component is presented in Chapter V. Finally, Chapter VI presents a design summary and recommendations.

II. Requirements Analysis

Expansion of the problem definition is twofold. First, the results of the requirements gathering phase of the thesis project are summarized. Second, the considerations that influence the development of the design are elaborated. Then before proceeding into the next chapter, an example of the actual game and the game flow is presented to provide a better understanding of the decisions made throughout the design and development process.

2.1 Requirements Summary

This section summarizes who the users are of the software developed in this thesis and their requirements for the microcomputer based BIG STICK wargame.

2.1.1 Definition of the User. The customer of the redesign and rehost effort is the Air Force Wargaming Center who runs the BIG STICK exercise once a year in the spring at the end of the ACSC nuclear warfare curriculum.

There are two primary operators of the actual BIG STICK system. One primary operator or user is the ACSC student. The student or more accurately, the team of students, spends approximately four days using the system, one day for each of force selection, deployment, targeting, and employment.

Another primary user is the programmer-maintenance personnel who must support the completed project through error correction, detection, and prevention; enhancement or modifications; and program optimization activities.

2.1.2 Customer Requirements. The customer requirements are to build a micro-computer based system that will:

- *Maintain the learning objectives of the BIG STICK exercise.* Keep the rules and flow of the game the same.

- *Use the BSPAT Force Selection and Force Deployment data files.* This may involve reimplementing if the current implementation is unusable. Use of BSPAT by the student operator must be optional.
- *Be more "user-friendly."* Make it simple and easy to use so that less time is spent on learning how to use the system. Make it easy for the students to enter, modify, save, and print entries. Provide immediate feedback when errors are made by the students.
- *Simplify the exercise manager's task of controlling the level of interaction between opposing teams.* This includes limiting unauthorized access to opposing team's data.
- *Simplify the programmer-maintenance personnel's support task.* Fully document the redesign and development of the simulation system and software.

2.1.8 Validation of Customer Requirements. The proposed system environment of Figure 1 was discussed with and accepted by the exercise staff.

A paper prototype of the game displays was then developed to further validate customer requirements. Login and force selection, deployment, targeting, and employment screen displays were presented to the exercise staff for approval. Development of an acceptable paper prototype was an iterative process as changes were made based on customer suggestions. Of vast appeal was the proposed use of screen oriented input entry and editing, on-line help, and macro keys to control movement between displays. After the paper prototype was implemented, a hands-on examination of the system by the staff was conducted. No new requirements were revealed during this examination.

The simulation program requirements were validated by a walk through of flowcharts developed for the game. Deficiencies were identified and corrected.

Final validation of current customer requirements was to be conducted during the product testing phase. Due to the time constrained nature of this project, full product implementation and testing was not completed.

2.2 Design Considerations

Some factors considered during the design phase include the characteristics of the human operators and the software and hardware requirements. The characteristics of the student and programmer-maintenance operators impose limitations on the development of the design. Software compatibility and hardware availability, which primarily limit the implementation of the project, do impose some restrictions on the design.

2.2.1 Characteristics of the Student Operator. Simpson describes four types of human computer operators: (1) computer professionals, (2) professionals without computer experience, (3) naive users, and (4) skilled clerks [19:22-23]. In general, the ACSC students are type two operators, professionals without computer experience. Characteristics of this type of operator described by Simpson basically hold true for the ACSC students and are as follows:

- They are intelligent and well-educated.
- They lack patience.
- They set high standards for program performance.
- They are intolerant of program errors.
- They know little about computers.
- They are not interested in knowing about computers and may not even like them.
- They know how to turn the computer on.
- They cannot be expected to remember anything that is not presented within the context of the program.
- They will consistently ignore screen prompts and will enter data that have inappropriate type, format, length, and other characteristics.
- They are motivated to accomplish the function the program was designed to serve.
- They resent it when things go wrong.

One other defining characteristic of the ACSC student operator is the limited exposure, approximately four days, to the BIG STICK system.

2.2.2 Characteristics of the Programmer-Maintenance Personnel. The current staff of wargaming programmer-maintenance personnel have technical training and computer experience. These computer professionals, or type one operators, care less about user-friendly features than about finding ways to speed up their use of the program. They are not intimidated by software, and they are familiar with software design concepts [19].

2.2.3 Software Limitations. The INGRES database management system is used in several applications such as the Theater Warfare Exercise (TWX) and the Joint Planning Exercise (JPLAN) at the Wargaming Center. FORTRAN is also supported. However, the PC Version of INGRES currently does not support the FORTRAN language. It does support the C language, which has been used at the Wargaming Center [11].

2.2.4 Hardware Limitations. Hardware facilities available to run the BIG STICK simulation are the Honeywell H6000 mainframe, Cybers, the Zenith Z-158 microcomputers with single and double Iomega Bernoulli disk cartridges. Printers are available to print force lists and damage reports.

2.3 The Game

After completing the planning portion of the BIG STICK exercise, ACSC students are able to begin the simulation or actual game play. The game consists of four parts: selection, deployment, targeting, and employment.

2.3.1 Force Selection. Each team may select or purchase *lots* of conventional and nuclear forces and defense systems. A *lot* consists of one or more units of a weapon or defense system. Enough tankers and weapons stores should be selected to support the offensive force. Teams may also elect to conduct intelligence operations, but must weigh in the cost factor. Teams must continue operation of one anti-ballistic missile (ABM) system and continue contribution for the operation of the Damage Assessment Satellite System (DASS). Force selection inputs are made to the computer by indicating the number of lots that are to be selected for each system identification number. Restrictions apply to the number of selected lots and the maximum and minimum budget expenditures. Lots not

selected are considered to be eliminated from the force structure; however, certain nuclear force systems may be placed into non-operational status at a reduced lot cost.

2.3.2 Force Deployment. Once selection inputs have been validated by the computer and saved at the request of the student operator, the weapon and defense systems can be deployed. Planning document (PD) forms are used to help students with the deployment phase. Each force system has a PD form with basing sites identified and instructions provided on how to indicate deployment of a lot or lots to a site. Every lot must be deployed to a site (or sector for anti-submarine warfare (ASW) task forces and sea-launched ballistic missiles (SLBMs) on submarines deployed at sea). Deployment restrictions are force system dependent and apply to the list of possible sites and the maximum number of lots. Students are allowed the freedom of specifying additional air defense bases. These entries are validated and saved before the start of the next phase.

2.3.3 Force Targeting. Each unit of each lot of each offensive force system may be targeted against a site. If the unit carries several bombs or missiles or if the unit is a multiple independently targeted reentry vehicle (MIRV), each bomb, missile, or warhead can be targeted. This is an information intense phase due to the large number of targetable units (approximately 400 per team), targeting sites (approximately 600 per team), and system routing and range requirements.

Each unit is identified by a distinct tail number. Each target site is identified by a distinct combination of sector, base, site/silo, and SAM (surface-to-air missile) battery numbers. Routing and range requirements restrict the force unit from routing through or targeting certain sectors. The target/option planning worksheet is used for targeting the entire force. For all units, students specify a target site and up to four employment options. Ingress and egress sectors and recover theaters are specified for particular systems. As in the other two phases, validation checks are made and final inputs are saved before employment begins.

2.3.4 Force Employment. The employment phase of the wargame represents the first 12-15 hours of a nuclear exchange and is fought in up to 17 time periods 12 of

which are exchange time periods, two are designated as retargeting periods, and three are available as negotiation periods. Every exchange period begins with a "Course of Action" query. If an option other than "No Action" is selected for the current exchange period, war is conducted and the simulation will normally proceed through four stages. The first stage is the launch stage. Second is the enroute stage which is followed by the impact stage and the report stage. Figure 3 shows a simplified flowchart of the different stages. Depending on the action selected in conjunction with the type of force structure available, the simulation Monte Carloes a series of probability checks such as an in-commission check to launch; a check for no-abort by aircraft or a reliability check for missiles followed by a check for avoiding area and point defenses during the enroute stage; then a check for damage during the impact stage. If the random number is less than or equal to the expected value probability, then the event is a success.

Retargeting periods, occurring at the end of exchange periods four and nine, are designed to allow players the option to retarget some of the available assets. All entries must be completed within 45 minutes.

Decisions to negotiate occur prior to the start of periods two, five, and ten after any retargeting is made. If it is determined that the teams will negotiate, then the next employment period is delayed for ten minutes while negotiations occur. These periods allow players to "exercise coercion, renegotiate rules of engagement, control escalation, and establish intrawar deterrence" [2:7-8]. In the negotiation process prior to period ten, teams may decide to terminate the war activity. However, at the start of period ten, a team or both teams may individually decide to continue the war since there is no requirement to adhere to the negotiated agreements.

2.3.5 Example of the Game. To demonstrate how the simulation proceeds, an example of a Blue bomber carrying four bombs and four air-to-surface missiles (ASMs) is selected to depart from the U.S. targeted for the U.S.S.R. According to the rules of the game, since this is a targeted bomber, it is on ground alert at the beginning of the game, and if it passes its in-commission check, then it is launched for survival and remains airborne over friendly territory until directed to enter enemy territory. Non-alert or non-

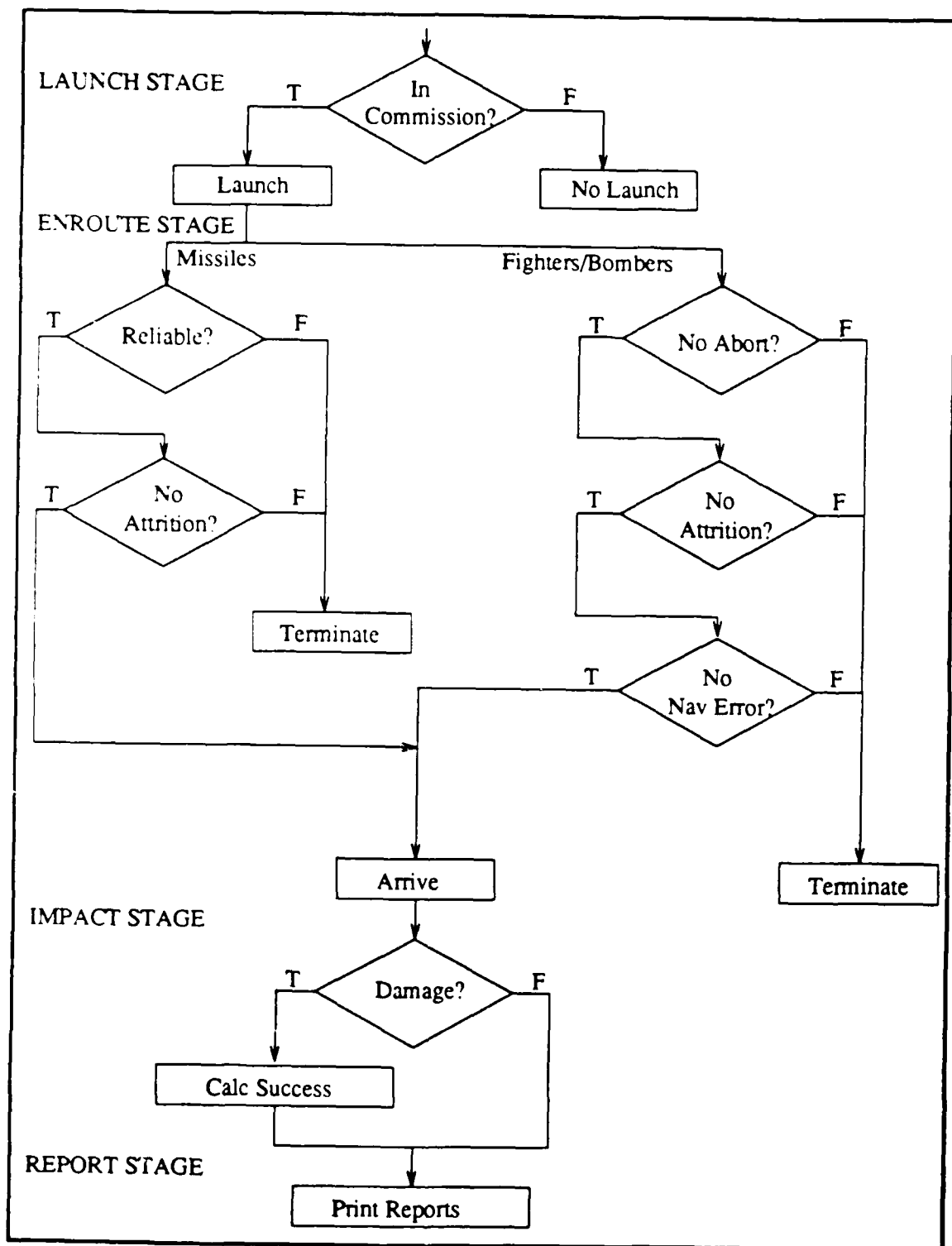


Figure 3. Simplified Flowchart of Game Play

targeted bombers are launched for survival and remain safe in orbit over friendly territory throughout the game, but must land at any undamaged airfield within its launch sector at the end of the game. If all airfields are damaged, then the bomber crashes. Bombers not launched remain on the ground and are susceptible to attack throughout the game.

Once a message has been sent from the National Command Authority directing the use of bombers against the enemy, the blue team could select the action "Execute Options", "Execute Individual Vehicles" or "Execute Both". The blue team selects to execute this bomber in time period two. This bomber and all other bombers can never reach enemy territory until after time period four to simulate travel time. In time period five the bomber enters the enroute stage and is checked for no abort (probability of no abort). Assuming no abort, then the bomber enters the first of possibly six area sectors and passes through area defenses controlled by enemy airborne warning and control system (AWACS) aircraft, interceptor aircraft, and Air Defense Control Centers (ADCC) deployed in the sector. If the bomber passes this check (probability of penetration), any ASMs targeted for sites in the enemy sector are launched and are no longer subject to area or point defenses. Each bomb targeted for a site within this sector is dropped assuming that the bomber finds the target (probability of no navigation error) and survives point defenses (probability of avoiding a SAM). Each time the bomber attacks a target it is subject to enemy point defense. Figure 4 illustrates the bomber simulation profile.

After the impact stage, the simulation proceeds to the fourth stage of the time period and generates a report on the success or failure of the bomber strike attack. This report does not reach the players until period eight to simulate delay and partial inaccuracy of reconnaissance data. This ends time period five. After this time period, the bomber having succeeded, will continue to the next enemy sector where it will once again be subject to area and point defenses, or the bomber may return to friendly territory depending on what was programmed to occur next.

Upon completion of its mission the bomber must return to friendly territory and land at an undamaged airfield else it will crash and be deducted from the end of game surviving assets report.

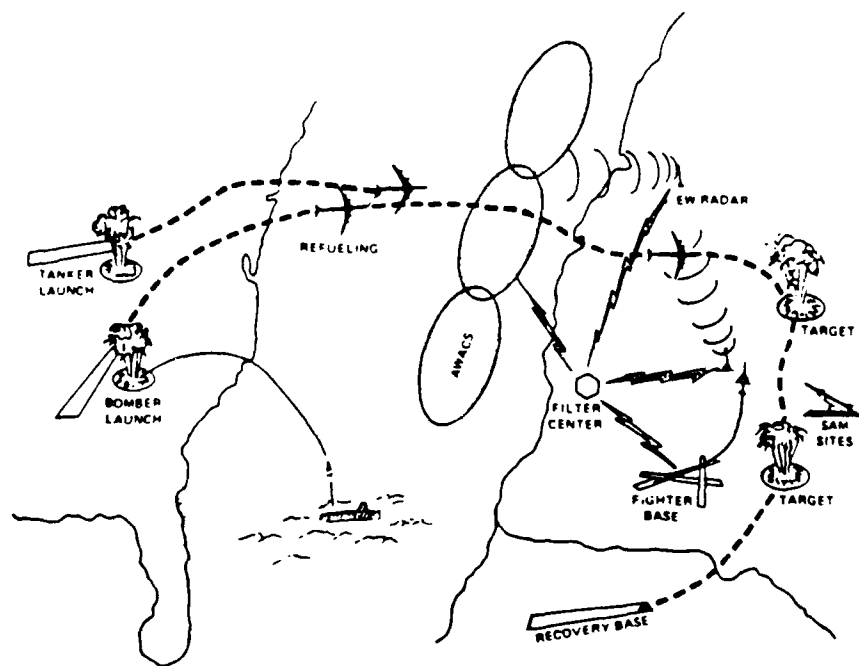


Figure 4. Bomber Simulation Profile [10]

III. Theoretical Development

This chapter presents a review of relevant portions of existing theory on which the design of the BIG STICK redesign and rehost effort is based. Areas reviewed are software engineering, database design, and user-friendly program design. In general, the design theories presented are discussed by Jansen [11] and Kross [14] in their design of similar efforts.

3.1 Software Engineering Approaches

The goals of software engineering are modifiability, efficiency, reliability, and understandability [4]. The effort to redesign and rehost the BIG STICK exercise is indeed a software project which can be engineered to satisfy these goals. This requires that a software engineering approach be adopted.

Prevailing approaches to software engineering are the classic life cycle, prototyping, and fourth generation techniques. Other approaches are usually hybrids of these models [16].

3.1.1 Life Cycle Model. The classic life cycle or waterfall model consists of iterations through systems engineering, analysis, design, code, testing, and maintenance phases. Theories and techniques specific to each of these activities have been developed and are widely used. Some of these supporting techniques are used in the thesis and are discussed later in the design and implementation and testing portions of this thesis.

The life cycle model has been adopted as the standard for most software development projects because of its structured approach to software development; however, it has been and still remains under much criticism. The model is accused of being too rigid to meet changing user requirements [3,15] and is often the source of late, incomplete, and error prone software [9].

3.1.2 Prototyping. This model is comprised of requirements gathering, quick design, building the prototype, evaluation and refinement of requirements, and product engineering phases. Prototyping provides an effective means for identifying and validating

user requirements. Like the life cycle model, it too is under criticism for encouraging development of less-than-ideal, compromised systems [16].

3.1.3 Fourth Generation Techniques (4GT). 4GT is another iterative approach. The phases of this approach are requirements gathering, design strategy, implementation using a fourth generation language (4GL), and product engineering. Use of a 4GL allows the developer to specify what is to be done instead of how it is done. Source code is then automatically generated from the specification. Advantages of this method are reduced development time and improved developer productivity, while disadvantages include difficulty in using 4GT tools and the inefficient source code.

3.1.4 Hybrids. The strengths of each of the primary models can be effectively combined and tailored to specific software engineering needs. As with the three primary models, all derived approaches should be structured and contain analysis, design, implementation, integration, and testing activities. Whether these activities are combined into four phases or distributed between eight phases, they must exist.

Boehm's spiral, risk driven model [3] is an example of a hybrid that is gaining a great deal of recognition. Each spiral in Boehm's model represents separate software components or increments and undergoes risk or needs determination, prototype, design, development, integration and test phases. Its primary advantage is that it accommodates the evolving needs of the user while using the strengths of the prevailing models mentioned earlier. One difficulty lies in obtaining risk assessment expertise.

3.2 Database Design Approaches

The force targeting and force employment discussions from the last chapter introduced the kind and amount of data, pieces of information, required by the BIG STICK simulation. This data has been stored in a large fixed, hard-coded file system. File systems in the past have been plagued with data redundancies and inconsistencies and with difficulties in accessing data. File systems also made enhancements to an application difficult. These problems led to the development of database systems [12].

A database is a collection of data organized into a single integrated, logical structure known as a schema [17]. A properly designed database attempts to remove or at least control data redundancy, does not allow data inconsistency, and allows easier access to the data items. A properly designed database will also ensure that all data items are correctly represented, that all relationships are correctly represented, and that all reports are supported [17]. Data items are represented by a set of attributes. The relationships are identified by the number of records in one set of data items that can be associated with a number of records from another set of data items. These relationships are one-to-one, one-to-many, many-to-one, and many-to-many.

Database design models, developed since the late 1960s, include the hierarchical, shallow network, relational, CODASYL network, and extended network or post-relational models. These models differ in the way the relationships between data items can be represented. The latter approach is a hybrid which integrates features from both the relational and the CODASYL network approaches.

The hierarchical model creates a tree-structured schema in which the relationships are represented by records and links.

The network models create an arbitrary graph schema in which relationships are also represented by records and links, but are less structured than the hierarchical approach. The various network models differ in the treatment of the one-to-many relationship.

The relational models, related to the mathematical concept of relations, rely on redundancy of key attributes of data items to represent relationships. All attributes are stored in tables called relations. As a result of the redundancy of key attributes, a process called normalization is required to ensure that unnecessary repetition of data is eliminated. The normalization process also preserves losslessness of information and allows the one-to-many relationship to be represented which would not otherwise be represented.

3.3 User-Friendly Program Design Principles

The BIG STICK customer desires a user-friendly system. User-friendly refers to the interface between the human operator and the computer. A *friendly* interface is one that

includes factors that recognize "memory, ability to see and hear, intelligence, motivation, motor skills" [19] and other human factors. In defining a friendly interface, Galitz starts with a dictionary definition and then goes on to describe desirable qualities of a user-friendly program. Some of these qualities are: adaptiveness, transparency, comprehensibility, naturalness, predictability, responsiveness, and forgiveness [7]. He also discusses ease of use as a design goal of an effective interface.

Most authors agree that the design of an effective, friendly interface is still an art. However, they do provide general principles and context-specific guidelines which together with the knowledge of the capabilities and limitations of the user can be used to develop the desired friendly interface.

The designer must first define the users and understand the capabilities and limitations of the users. The designer must anticipate the environment in which the program will be used. The designer must also [7,18,19]:

- *Give the users control.* Allow users to set the pace. For naive users, dialog should be initiated by the computer, and for experienced users, dialog should be initiated by the user.
- *Minimize the users' work.* Make efficient use of hand and eye movements.
- *Keep the program simple.*
- *Be consistent.* Prompts, commands, warnings should be consistent. Information coded in the display by use of color, brightness, flash rate and sound should be consistent. Prevention, detection and correction of errors should be consistent. Use of function keys should be consistent.
- *Give adequate feedback.* Be responsive by acknowledging all actions immediately through execution, in-progress messages, correction messages, and confirmation messages.
- *Not overstress working memory.* Short-term memory has a capacity of about 7 ± 2 items and holds on to information for about 15 seconds [19].
- *Minimize dependence on recall memory.*

- *Help the users remain oriented.* Carefully structure the program; provide a map of the structure; and identify the location of the operator within the structure.
- *Code information appropriately (or not at all).*
- *Prevent, detect, and correct errors.* Be forgiving and allow users to return immediately to a certain point if difficulties arise without the need to complete current action.
- *Use language that is natural, concise, informative, and discriminating.* Use short, simple sentences. Make statements positive versus negative. Use active rather than passive voice.
- *Provide assistance.* Assistance should be in the form of user-selectable prompting and a help facility.

3.3.1 System Security. Security is addressed for two reasons. First, the customer has identified a requirement for limiting access to data loaded on opposing teams' computers. Second, "user-friendly means abuser-friendly" [5:12-13]. Cronin claims that programs intended to lead users by the hand through an application, do not distinguish between authorized and unauthorized users.

Prevention or control of unauthorized access include limiting physical access to the computer, using passwords to limit access to the program and database, and unloading critical information and locking the system [13].

3.3.2 User's Manuals. Documentation is an important part in making the program user-friendly. In designing the user's manual as with any other written document, the audience or user must be defined. The user's manual should be based on actual user dialog and illustrate the use of the system in action by showing actual display sequences that allow the user to achieve desired objectives [18]

3.4 Summary

Several approaches to software engineering, database design, and user interface design were presented. Each of these approaches are applicable to the redesign and rehost of the

BIG STICK simulation; however, only one approach for each component was selected in this thesis. The selected design approaches, the reasons for the decisions, and the actual component designs are presented next.

IV. System and Component Design

The BIG STICK system was designed and developed using the 4GT development model, modified to include prototyping (see Figure 2). The advantage of this model is that it provides a simple four phase structure that allows the INGRES fourth generation tools such as the forms tool and 4GL programming language to be used. Use of these tools eliminates some of the coding requirements of the implementation phase and makes changes to the system easier. The flexibility in the product phase allows the final system to be engineered or made ready for use as each component is completed.

As part of the overall design strategy, the BIG STICK system was separated into six design components. Each component was designed, as necessary, using principles and techniques unique to that component.

Design and implementation of the database control system component was already developed and provided through the INGRES software packages. Design of the report component coincided with the development of the user interface design, since the BIG STICK reports are simply printed copies of the summaries displayed to the terminal. The design and implementation of the interconnection between the two end users was limited to floppy disk swapping, since software and hardware capabilities for a more effective means of interconnection are currently not available. Thus there is no real need to discuss issues for these three components.

The remainder of the discussion in this chapter is dedicated to the development of the database, user interface, and simulation program component designs. Major decisions and considerations made in the development of these designs are presented.

4.1 Database Design

Discussion of the database design includes choice of the database design model and development of the design. Exceptions to the database design, the decision to partition the database, and descriptions of how the design diagrams are reduced into tables and the database is normalized are also presented.

4.1.1 Choice of Database Design Model. The relational model was adopted as the design model for the BIG STICK database for the following reasons:

1. The relational database meets customer requirements and the software and hardware limitations.
2. The relational model can be directly implemented from the entity-relationship (E-R) diagram which provides a pictorial understanding of the database structure.
3. Entities and relations, similar to objects and operations, are easier to understand than records and links.

4.1.2 Development of the Relational Database Design. The design of the BIG STICK relational database was developed using the E-R diagram. The first step in developing the E-R diagram was to identify all BIG STICK data items or entities. One major data item is the site entity. Its attributes are identification number, name, type, composition, and status. The other major data item is the force system entity labeled *force_sum*. Most other data items were subtypes of these two entities. The E-R diagram in Figure 5 shows the two main entities, represented by boxes, and their subtypes represented through ISA links. Attributes are not shown.

Entities can be associated with other entities. These associations, called relationships, are represented by diamonds. Every entity need not be related to any or all other entities. Figure 5 shows some entities as stand alone entities, and it also shows that the two main entities are not joined.

After identifying all entities and relationships, the mapping constraints (one-to-one, one-to-many, many-to-one, and many-to-many) were identified. All relationships in Figure 5 are many-to-many relationships.

The double boxed entities represent weak entities. These exist only if the strong entity with which they are associated exists. Figure 5 shows five weak entities. These entities contain the legal targeting sectors of the various weapon system subtypes. If the strong weapon system subtype entities did not exist, then neither would the weak entities.

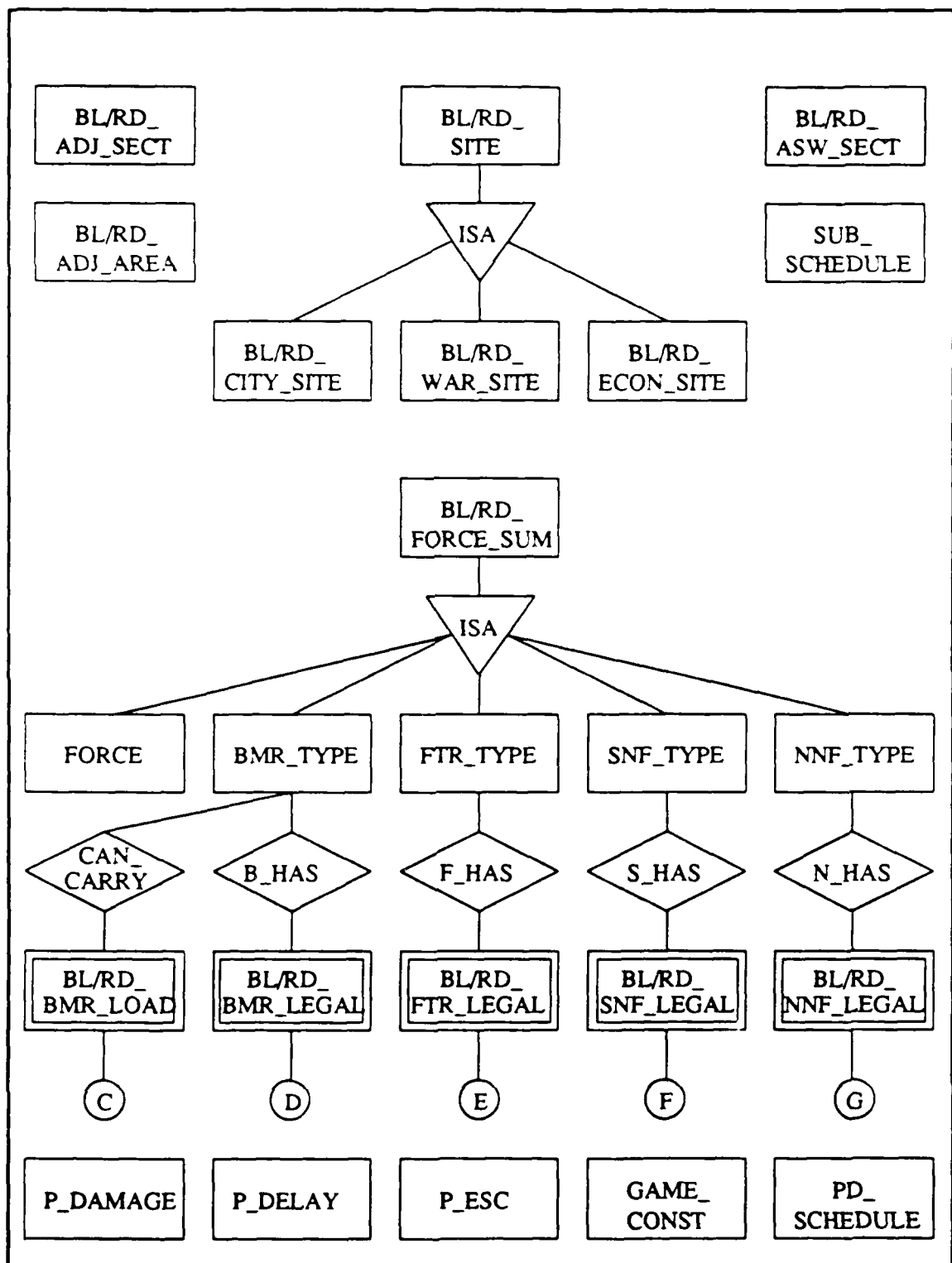


Figure 5. BIG STICK E-R Diagram

The E-R diagram in Figure 6 shows the entities and relationships required to proceed with the force selection and force deployment portions of the BIG STICK simulation. It further expands the site entity of Figure 5. Examples of one-to-one and many-to-one relationships are represented in the Figure 6 diagram.

The E-R diagram of Figure 7 expands the force entity of Figure 5 and shows the entities and relationships required to proceed with the force targeting and force employment portions of the simulation. The small labeled circles represent connections between entities shown in separate figures.

4.1.3 Database Exceptions. BIG STICK information provided for completeness, but not required by the simulation, was not included in the database. These items are missile alert rates and certain probabilities whose values are one, such as PABM (probability of avoiding an ABM) for all sectors not containing an ABM site.

Although targeting range values are directly represented as an attribute of the force subtypes, these values are also encoded in the legal targeting sectors entities of Figure 5. For example, the range for the non-nuclear force (nnf) missiles is two. The legal targeting sectors entity for non-nuclear force missiles, *nnf_legal*, contains all combinations of deploy sectors and targeting sectors that are within the two sector range. This structure was selected because there were often as many exceptions as there were possible legal sectors. This structure also requires a single search condition to determine the validity of the targeting specification.

4.1.4 Partitioning of the Database. Most entities and relations had *side* as an attribute. The value of *side* was either blue or red. To meet possible memory and storage space constraints of the microcomputer based system, the database was partitioned according to *side*. This, in effect, creates two databases: one for the blue side and one for the red side. Although there are more tables to manage, each table now has one less attribute.

An example of the partitioning is the site entity. Originally it had *side* as an attribute. The site entity is now *bl_site* and *rd_site*. Entities not having *side* as an attribute are included in both databases.

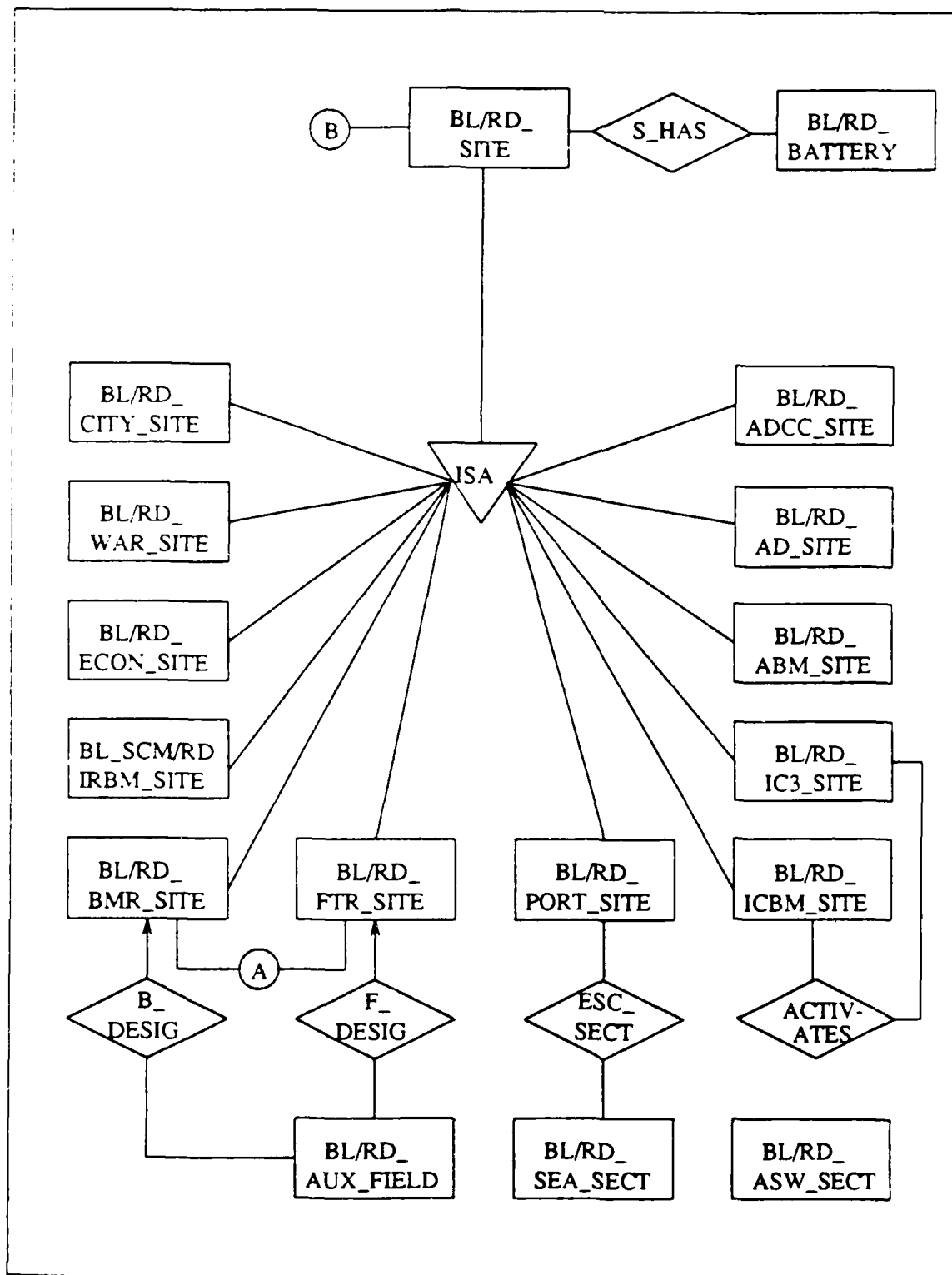


Figure 6. Site Entities and Relationships

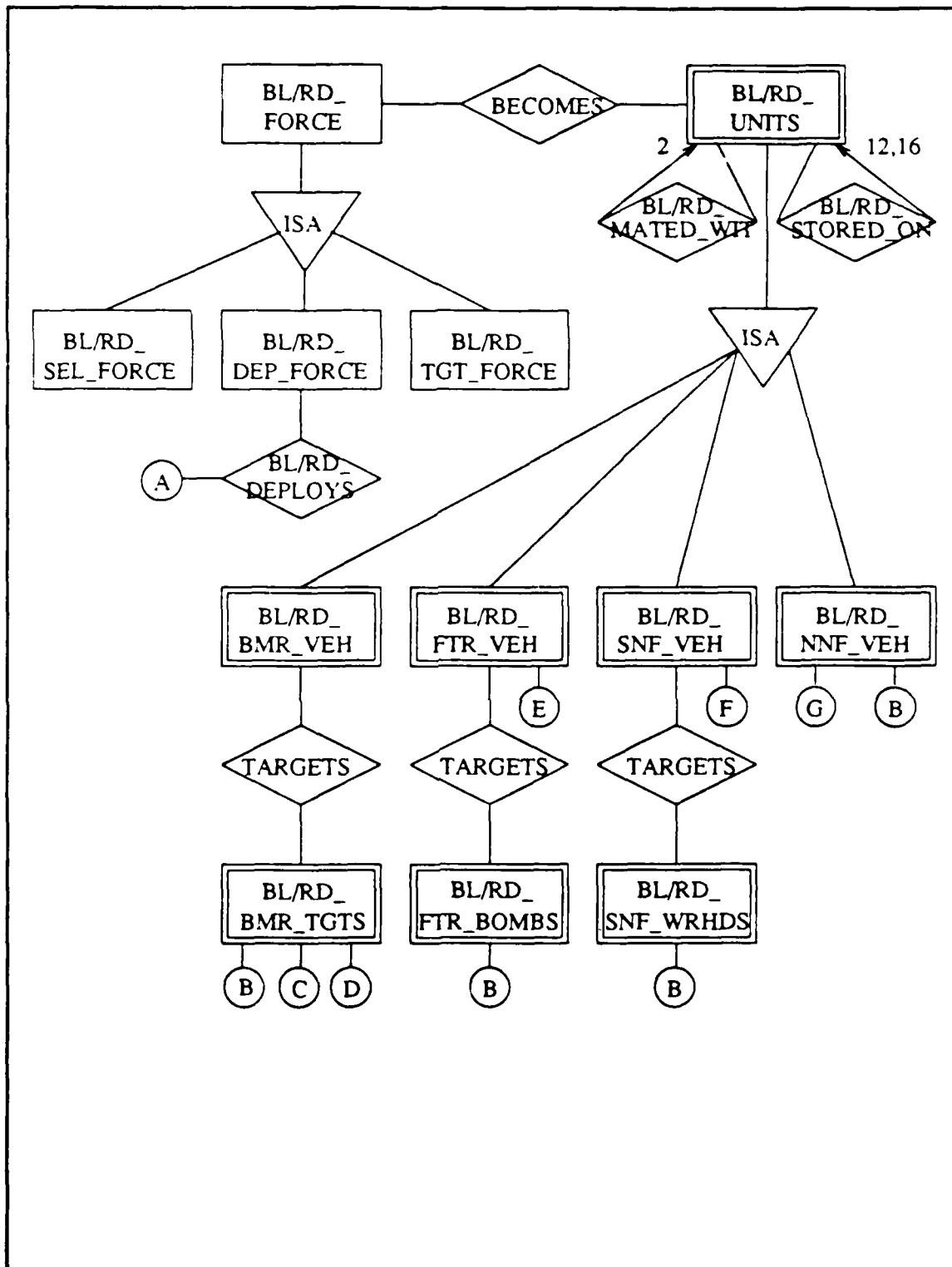


Figure 7. Force Entities and Relationships

4.1.5 Reducing E-R Diagrams to Tables. Tables called relations are used to store the attributes of each entity and relationship. Each table must have an identified key which uniquely identifies every row in the table which represents an instance of an entity in the entity set. Strong entities have primary keys which may consist of one or more attributes. Weak entities inherit the key attributes of their related strong entity. The relationships surrounding a weak entity are not reduced to tables; however, all other relationships are reduced to tables and inherit the key attributes of their related entities.

4.1.6 Transforming ISA Links into Tables. Korth and Silberschatz [12] describe two ways of transforming ISA links into tables. In general, the decision was made to design the database using the first method of creating a separate table for the higher level entity, and at the lower level, creating tables that include only the key attributes from the higher level. This creates more tables with less attributes per table. The alternate method eliminates the higher level table and creates bigger lower level tables. Besides the advantage of having smaller tables, the first method provides a more logical structure of the database.

4.1.7 Normalization. Prior to database implementation, the relations were normalized. Normalization requires that all functional dependencies or the dependencies that exist between key and non-key attributes be identified and any redundant dependencies be eliminated. Since each relation had only one functional dependency, that of the non-key attributes with the primary key, no further decomposition was required. Final key and non-key attribute names and format types for the BIG STICK database relations are presented in Appendix B. Key attributes are marked with an asterisk.

4.2 User Interface Design

A single interface design was developed for both sides. Developing a single interface reduces the amount of overall system code and is easier to maintain than maintaining code for two separate interfaces. A disadvantage of a single interface is presented later in the interface implementation discussion of Chapter V.

Design began with the preparation of sample screen layouts. Security was also designed for in this initial effort. Once the layouts were approved, other design steps were taken to prepare the layouts for implementation.

4.2.1 Choice of Screen Design. In the past, the student users needed to transfer force selection, deployment, and targeting decisions from the BIG STICK planning document (PD) forms to a computer input form. This computer input form aided students in formatting entries with appropriate commas and spaces for input to the Honeywell system. However, in the new system, the choice of screen design was a direct implementation of the BIG STICK planning document (PD) forms, eliminating the intermediate translation step.

4.2.2 Development of Screen Design. Paper samples of the screen layouts, or forms, including menu options were developed near the start of the project as a method of validating user requirements. An attempt was made to keep the screen layouts simple and consistent with respect to appearance and function. Figure 8 shows one screen layout from the original prototype.

In each screen design a standard set of function keys were used and are as follows:

<i>Key</i>	<i>Description</i>
------------	--------------------

F3	Save
----	------

F4	Print
----	-------

F5	PrevNext (System, Tail Nr, or Sector)
----	---------------------------------------

F6	Proceed to One Level Down
----	---------------------------

F10	Quit to Next Level Up
-----	-----------------------

Function key [F2] is a special INGRES key which when selected displays the explanation for the function key highlighted in the menu. Function key [F8], another special key, temporarily returns the user to the operating system. Use of these special keys are not required in the simulation.

4.2.3 System Security. Prevention or control of unauthorized access using passwords was the only security measure designed for in this effort. Students would need to

***** INSERT/MODIFY SELECTED FORCES (BLUE) *****

START-V STRATEGIC FORCE SUMMARY - PD FORM 2

SYS ID	SYSTEM NAME	UNITS/ LOT	COST/ LOT	TOTAL LOTS	TOTAL COST
1	INERCEPTORS	2	0.475		(Determined by System)
2	AWACS	1	0.7		
3	ADCC	1	1.1		
4	ABM	18	15.1	1	15.1
5	SAM	5	0.3		
6	SAM-D	10	0.5		
7	ASW	1	2.0		
8	CIVIL DEF	1	0.3		
9	B-1	4	3.65		
10	B-2	4	2.5		
11	TANKER	1	0.6		
12	ICBM-I	9	3.3		
13	ICBM-II	9	2.5		
				TOTAL: (Determined by System)	

F1 - Help F2 - Print F3 - Save F10 - Return

Figure 8. Sample Prototype Screen Layout

enter the correct combination of seminar number, team number, and password to access the main menu of the game.

4.2.4 Other Interface Design Steps. Other design activities that took place prior to implementation of the user interface include the identification of frames and forms. As described earlier, an INGRES frame consists of forms or the actual screen layouts along with menu options located at the top or bottom of the form. Each frame uses a form which may have the same name as the frame. Names and formats of the simple data fields and table data fields used in each form were identified.

Applications, or sets of frames, and the flow of control between applications were designed. The subgrouping of the frame sets into "small" applications was required to prevent problems with memory constraints previously encountered in other PC INGRES database developments [11,14]. Definition of "small" was not known during the design phase thus final description of the application design could not be complete until the implementation phase. The final frame application design is presented in Figure 9.

Other design decisions in conjunction with considerations of the capabilities of the PC INGRES Forms tool and 4GL programming language were made during the actual implementation of the frames. The interface was a menu driven system which gives the student users control over the actions and the pace at which they prefer to work. Options included in each of the main menu frames were identified using single letters corresponding to the first letter of the choice. For example, to select Force Selection from the main menu, the student would select 'S' or to select the Modify Selected Forces option, the student would select 'M'. This helped minimize the users' work and minimize dependence on recall memory. Use of function keys to activate the menu options in other frames also helped in reducing the number of strokes required to accomplish a certain action.

Feedback was designed for in several ways. In-progress messages are to be used wherever a transition is made between applications and between large frames. In-progress messages are also used during a database query to a large database table that could take a noticeable amount of time. Error and correction messages were also included in the interface design.

Prior to the display of error and correction messages, error detection must occur. Detection was designed for in the interface. Possibilities for student input errors are numerous regardless of whether they are deliberate or unintentional. Management of as many possible error situations as could be anticipated were included in the design. Responses would be provided as soon as an error was detected.

To help students avoid making errors, a help facility provided via selection of a function key option. The [F1] key is used consistently in each of the frames to activate the help facility. The help facility helps the students remain oriented and also provides information on the rules or constraints of the BIG STICK exercise particular to the frame and on what the students are able to do within the frame and how to go about doing it.

Color was used to code distinct types of information presented in the frame. For example, each frame containing information unique to the team (this excludes frames such as the main menu display) also includes a display field called *side* whose value is either "Blue" or "Red". This field was colored accordingly. Another example of the use of color was to separate deployment information from execution options and a separate color for targeted sites. Actual color decisions were made by trial and error selection from the six colors available on the Zenith color monitors.

Information was also coded using brightness and reverse video displays. Reverse video was used to highlight user input fields. Brightness was used to highlight important information contained in display-only fields.

4.3 Simulation Program Design

The design of the simulation program was developed using top-down, functional decomposition. Tools used in the development of the design were IDEF₀ diagrams and flowcharts.

IDEF₀ or identification language, level 0, developed for the Air Force by SofTech, Inc. is equivalent to their SADT diagrams. First, a top level diagram with input, output, control, and mechanism parameters for each module is drawn. A top level diagram for the BIG STICK simulation program design is presented in Figure 9. The modules in this

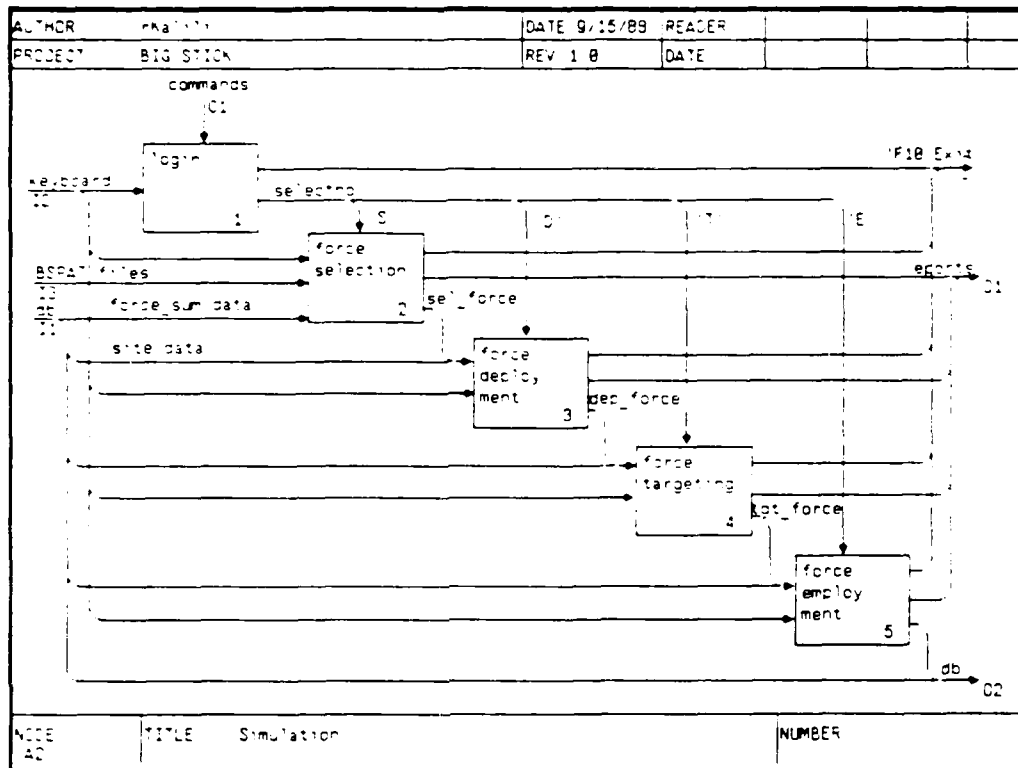


Figure 9. A2-Simulation IDEF₀ Diagram

diagram are then functionally decomposed and diagrammed at each level of decomposition until all requirements are identified and represented at a suitable level of detail. Remaining design IDEF₀ diagrams are contained in Appendix C.

Portions of these diagrams were then translated into flowcharts, which are graphical means of representing the order or sequence of activities. These flowcharts are similar to the one presented in Figure 3 except that the decisions and processes are more specifically detailed. These flowcharts contained in Appendix D were reviewed by the BIG STICK exercise staff and validated against the old FORTRAN code.

V. Implementation and Component Testing

In order to effectively manage control over the large development effort, the BIG STICK simulation was implemented incrementally by component beginning with the database. The user interface forms and frames were developed and the report specifications were built next. Finally, some maintenance programs were created to allow the programmer-maintenance personnel to modify the database as necessary.

5.1 Database Implementation

Implementation of the database was straight forward. The relations identified in the design phase were used to create the table specifications in INGRES using the Query-By-Forms (QBF) package. The table specifications for the blue site entity and force summary entity are presented in Tables 1 and 2. These tables show the names and format of each column attribute in the database tables.

These tables and the tables for all other entities and relationships displayed in Figure 5 were built using QBF. Data was entered for these fields also using QBF. Tables for entities and relationships in Figures 6 and 7 are not created until they are needed for the force selection, deployment, targeting, and employment phases of the simulation. These tables are created using the SQL create, insert and update commands of the INGRES-provided database control system. The SQL commands are placed in procedures used to initialize each game phase. Additional tables not included as part of the original design such as the *login* table were created and destroyed as needed to support the simulation program.

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
site_name	vchar(13)
site_type	vchar(4)
hd_sof_type	vchar(1)
site_status	integer1

Table 1. Blue Site Table Description

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
sys_name	vchar(12)
lot_units	integer1
lot_cost	float(4)
pd_fm	vchar(6)

Table 2. Force Summary Table Description

No significant problems were encountered in the implementation and component testing of the database.

5.2 User Interface Implementation

Interface implementation was two-phased. First, the forms were built from the approved layouts containing the form and field names derived during the design phase. Then the 4GL program code that fills the form, builds the menu options, and controls the movement between forms was developed to complete the frame.

5.2.1 Forms Development. The INGRES Forms tool called Visual Forms Editor (VIFRED) allows the developer to specify screen layout and display features of the simple and table data field without writing any lines of code. Use of features such as reverse video, color, and brightness required that the developer toggle the feature on or off.

Final form display of the Force Selection layout presented earlier in Figure 8 is presented here in Figure 10. Display features in this screen, which cannot be distinguished in the figure, include color coding of the side "Blue" or "Red" and the reverse video highlighting technique in the column where the students are allowed to enter the force selection inputs. All other columns were display only fields and were not highlighted. In all forms, titles and tables were centered for a more pleasing appearance.

One minor INGRES system constraint imposed in the development of the forms was the single line column heading for table data fields. The width of a table field column is determined by the length of the data field or the column heading, whichever was larger. In general, the headings were much larger than the required width of its corresponding data

[F1] - Help [F3] - Save [F4] - Print [F10] - Quit						
Blue						
PD FM 2						
START-U STRATEGIC FORCE SUMMARY						
SYS ID	SYSTEM NAME	UNITS/LOT	COST/LOT	TOTAL LOTS	TOTAL UNITS	TOTAL COST
1	INTERCEPTORS	2	0.475	68	136	32.300
2	AWACS	1	0.700	5	5	3.500
3	ADCC	1	1.100	15	15	16.500
4	ABM	18	15.100	1	18	15.100
5	SAM	5	0.300	18	90	5.400
6	SAM-D	10	0.500	16	160	8.000
7	ASW	1	2.000	2	2	4.000
8	CIVIL DEF	1	0.300	19	19	5.700
9	B-1	4	3.650	4	16	14.600
10	B-2	4	2.500	4	16	10.000
11	TANKER	1	0.600	46	46	27.600
12	ICBM-I	9	3.300	1	9	3.300
13	ICBM-II	9	2.500	1	9	2.500
14	ICBM-III	9	2.800	2	18	5.600
15	SLBM-I	16	7.800	1	16	7.800
16	SLBM-II	12	9.700	2	24	19.400
TOTAL:						228.900

Figure 10. Final Force Selection Screen

field. A double line heading would reduce the width of the columns and the overall table size which would result in a symmetrical table display.

5.2.2 Menu Options Modification. One addition was made to the original set of function keys available as menu options. Within a table field the user could use the 'PgUp' key to scroll through the table a full display at one time. This did not work with the 'PgDn' key. Therefore, a separate function key [F9]-Page Down was implemented. However, because it was a slow procedure, use of the [F9] key was reduced to support only those frames in which no other scrolling mechanism was available.

The one frame which did require this scroll mechanism was the Force Deployment Summary frame. In the frame, all deployable forces are listed in a display-only tablefield. At the bottom of the frame is a simple field in which the user would enter the identification number of the system he or she wanted to deploy. Since the number of available force systems would generally be more than could be displayed on one screen and since the user

cannot not scroll through a display-only field, use of the [F9] key allows the user to bring to the display table the remaining force systems.

5.2.3 Frame Development The 4GL programming language was used to create the frame menu options, to fill the forms with information useful to the frame, and to control the movement between frames within and across applications. The 4GL modules are designed to activate the field or menu option instructions if an entry is made to a data field or if a menu option is selected.

After defining and creating an application and the frames within the application by using the INGRES Applications-By-Forms (ABF) package, the frames (code and forms) could be compiled and individually tested.

Development of the frames was much more complex than the implementation of the database and the development of the forms. Reasons for the complexity and ultimately the extended time required for the frame implementation were error management and the problem of memory constraints. One interface design principle was to prevent, detect, and correct errors. The possibilities for errors in all phases of the game were enormous because of the numerous restrictions imposed by the game rules. Although the 4GL code fully supported the error management requirements, the task of identifying and managing all possibilities remained complex.

5.2.4 Interface Implementation Problems. A problem previously encountered by Kross [15] and Jansen [12] and addressed by Relational Technology in the release notes for the INGRES Version 5.0/02a update was the memory limitations for 4GL applications. The "out of memory" problems occurred under the following instances:

1. When compiling single 4GL modules larger than 30,000 bytes;
2. When using the Visual Forms Editor (VIFRED) on a form that has a large number of fields and a large number of columns within a tablefield;
3. A combination of 1 and 2 above, when compiling a large module with less than 30,000 bytes with a medium-sized form;

4. When calling more than five levels of frames successively within a single application.

Suggested solutions to these problems are liberal use of "SET RELEASEMEM" commands after single large queries or a series of small queries, decomposition of large applications, reducing the number of buffers specified in the config.sys file or reinstalling INGRES with fewer pages in the buffer manager with the "-Bxx" and increasing the dynamic memory with the "-Dxxxxx" command.

Added use of the "SET RELEASEMEM" command was implemented. Although the applications were decomposed into small applications during the design, some including single frame applications were not small enough, particularly the force targeting frames. The single frame applications were larger than the 30,000 byte size. The solution was to replace portions of the code by calls to procedures. An appropriate buffer size specified in the config.sys file and the "-Bxx" command was 20-23. Under these conditions, the "-Dxxxxx" command was not necessary. However, 64,000 bytes, previously recommended for the dynamic memory size, was used.

Another problem occurred when the database and interface applications were moved from the Zenith Z-248 to the Z-158 microcomputer. The working Z-248 model failed to execute properly on the Z-158. Applications could be executed individually, but when an application made a call to another application, the "application could not be opened; general execution failure" message appeared and further execution was halted. The problem was narrowed down to the Bernoulli disk cartridge since the applications did work on a different cartridge.

5.2.5 Interface Component Testing. A test plan was developed to systematically test a range of possible and erroneous inputs to the selection, deployment, and targeting phases of the simulation. The main technique used was boundary value testing. Similar analysis was conducted for both the blue and the red sides. The interface program errors discovered included the erroneous use of blue relations instead of red relations for the red side; failure to recalculate some accumulated totals properly; and failure to use enough feedback messages particularly during queries that unexpectedly took a long time

to process. After these and other errors and deficiencies were corrected, the component was tested again until the known errors were removed and no new errors were discovered.

5.8 Maintenance Program Implementation

The purpose of the maintenance program was to provide an efficient means for adding to, deleting from, and changing the simulation database. This program, to be used by the programmer-maintenance personnel, is menu-driven and simply calls the PC INGRES Query-By-Forms Package, which provides an efficient means for accessing and modifying the database. The program also calls the PC INGRES Report Writer Package to print out the data contained in the database relations. No significant problems were encountered in the implementation of the maintenance program.

VI. Design Summary and Recommendations

This thesis presented a background of the BIG STICK simulation and its old hardware and software environment. The old simulation environment required improvements to the user-hostile, difficult to maintain system. This project was developed to overcome these problems. Although full implementation of the microcomputer-based system, which was not completed in this project, is necessary to determine the actual success of the design, the primary goal of this thesis was met with success.

As a result of this thesis project, the BIG STICK simulation now has a new microcomputer database and a new, screen-oriented, user-friendly interface for the Force Selection, Deployment, and Targeting phases of the simulation.

6.1 Design Summary

The software engineering goals of software modifiability, efficiency, reliability, and understandability were met with success using the modified fourth generation techniques (4GT) approach to the system development cycle. Use of the PC INGRES database management tools such as the Visual Forms Editor (VIFRED) and the Applications-By-Forms (ABF) package and the 4GL language requires structured, modular application design which are good software engineering principles that also helped to meet the above stated goals.

Six components create the new BIG STICK simulation environment: the application program, the database, the database control system, the screen control system, the reports, and the interconnection between the two end users. Each component was developed individually under the overall 4GT development cycle. Detailed description of the database and user-friendly screen control system component design, implementation, and testing and the design of the application program component was presented in this thesis.

The database was designed using the relational model. Entities and relationships were identified and combined into the E-R diagram from which the relations or tables were derived. Implementation of these relations was accomplished and data was entered

to fill the database. Test of the database implementation consisted of random test queries using the interactive database control system.

The screen control system or user-friendly interface was designed according to various principles and guidelines concerning screen layouts and display features, message prompting, and error management. The primary goal in the development of the interface as well as the report specifications was to keep the design simple and consistent. Implementation of the forms representing the screen layouts and the 4GL code used to support the forms took some time to complete. Reasons for the delay were attributed to the complexity of error management and limitation on application size due to memory constraints.

Application program design was developed using top-down, functional decomposition. The requirements and design of the program was captured in requirements diagrams and structured flowcharts.

6.2 Recommendations

The first and foremost suggestion for further action is the completion of the simulation code followed by full system testing.

Another suggestion, provided hardware and software capabilities become available, is to implement a more effective means for transferring data between the systems used by opposing student teams. Possible alternatives are to use a third microcomputer or a workstation which could link directly to the student terminals provided there was a suitable control program that would complete the data transfer. Another alternative is to add a networking system. This would also require a suitable control system.

Enhancements can be made to the microcomputer-based BIG STICK simulation in the areas of performance and added game parameters. Porting the simulation to another microcomputer with a faster processor than the current 8088 would greatly improve the speed of the program run time performance. Finally, the addition of new weapon systems such as strategic defense space systems would demonstrate the advantage of having a database supported application while meeting new, evolving requirements of the Air Force Wargaming Center.

Appendix A. *New Deployment PD Form Numbering Scheme*

<i>New</i>	<i>Old</i>	<i>System Name and (ID)</i>
3	3, 4	Civil Defense (8)
5	5	ADCC (3)
6	5	ASW (7)
7a	6, 7	Aux AD Bases
7	6, 7	AD Bases, Ints & AWACS (1-2)
8	8, 10	SAM (5)
9	9	SAM-D (6)
10	11	ICBM (12-14, 28-30)
11	12	SLBM (15-16, 31-32)
12	13, 14	Bombers (9-10, 26-27)
13	15, 16	Fighters (19-20)
14	17	SCM/IRBM (21/22)

Table 3. New PD Form Schedule

Appendix B. Database Relations

The following tables contain the description of the database relations required by the simulation as identified in the Entity Relationship diagrams of Figures 5, 6, and 7. The relations are described by *Column Name* which is the name of each attribute and *Format* which specifies the data format of the attribute. Key attributes are marked with an asterisk.

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
tot_sel_lots	integer1
tot_sel_unit	integer2

Table 4. Blue/Red Select Force Relation

<i>Column Name</i>	<i>Format</i>
sector *	integer1
next_sector *	integer1

Table 5. Blue/Red Adjacent Sector Relation

<i>Column Name</i>	<i>Format</i>
area *	vchar(2)
sector *	integer1

Table 6. Blue/Red Adjacent Area Relation

<i>Column Name</i>	<i>Format</i>
sector_id *	integer1
asw_oper	integer1

Table 7. Blue/Red ASW Operational Sector Relation

<i>Column Name</i>	<i>Format</i>
num_subs *	integer1
subs_port	integer1
subs_sea	integer2

Table 8. Submarine Schedule Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
tgt_range	integer1
alert_rate	float4
p_ico	float4
p_sam	float4
p_nab	float4
p_nge	float4
p-pena	float4
p-penae	float4
p-penb	float4
p-penbe	float4
p-penc	float4
p-pence	float4

Table 9. Bomber Type Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
p_ico	float4
p_sam	float4
p_rel	float4
p-pena	float4
p-penb	float4
p-penc	float4

Table 10. Blue/Red Bomber Legal Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
load	vchar(1)
max_bombs	integer1
max_asms	integer1

Table 11. Blue/Red Bomber Load Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
max_bombs	integer1
min_bombs	integer1
alert_rate	float4
p_ico	float4
p_sam	float4
p_nab	float4
p_nge	float4
p_pena	float4
p_penb	float4
p_penc	float4

Table 12. Fighter Type Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
dep_sector	integer1
tgt_sector	integer1
rec_sector	integer1

Table 13. Blue/Red Fighter Legal Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
unit_wrhds	integer1
p_ico	float4
p_rel	float4
p_abm	float4
p_asw	float4
p_fix	float4

Table 14. Strategic Nuclear Force (SNF) Type Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
dep_sector	integer1
tgt_sector	integer1

Table 15. Blue/Red SNF Legal Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
p_ico	float4
p_sam	float4
p_rel	float4
p_pena	float4
p_penb	float4
p_penc	float4

Table 16. Non-Nuclear Force (NNF) Type Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
dep_sector	integer1
tgt_sector	integer1

Table 17. Blue/Red NNF Legal Relation

<i>Column Name</i>	<i>Format</i>
prob_name *	vchar(5)
num_destroy *	integer1
prob_val	float4

Table 18. Probability of Delay Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
hd_prob	float4
sof_prob	float4

Table 19. Probability of Damage Relation

<i>Column Name</i>	<i>Format</i>
time_period *	integer1
prob_val	float4

Table 20. Probability of Escape Relation

<i>Column Name</i>	<i>Format</i>
max_cost	float4
max_snf	integer1
pop_hit	float4
cap_hit	float4
option_r	float4
intrrate	float4
int1rate	float4
int2rate	float4
awacspico	float4

Table 21. Game Constants Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
batt_nr	integer1
num_sams	integer1
batt_status	integer1

Table 22. Blue/Red Battery Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
cd_prot	vchar(1)
pop_no_hit	float4
pop_hit_cd	float4

Table 23. Blue/Red City Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
cap_no_hit	integer1

Table 24. Blue/Red War Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
cap_no_hit	integer1

Table 25. Blue/Red Economic Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
num_miss	integer1
deployed	vchar(1)

Table 26. Blue SCM/Red IRBM Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
sac_wpn_stor	vchar(1)
stor_status	integer1

Table 27. Blue/Red Bomber Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
tac_wpn_stor	vchar(1)
stor_status	integer1

Table 28. Blue/Red Fighter Site Relation

<i>Column Name</i>	<i>Format</i>
sector_id *	integer1
site_id *	integer4
num_ints	integer2
num_awacs	integer1

Table 29. Blue/Red Auxiliary Air Field Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
slbm_type	integer1
slbm_2type	integer1

Table 30. Blue/Red Port Site Relation

<i>Column Name</i>	<i>Format</i>
port_id *	integer4
patrol_sect	integer1

Table 31. Escape Sector Relation

<i>Column Name</i>	<i>Format</i>
atseanr *	integer1
patrol_sect	integer1
sub_type	integer1
sub_status	integer1

Table 32. Blue/Red Sea Sector Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
icbm_type	integer1
silo_status	integer1

Table 33. Blue/Red ICBM Site Relation

<i>Column Name</i>	<i>Format</i>
icbm_site_id *	integer4
ic3_site_id *	integer1

Table 34. Blue/Red Activates Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
active	integer1

Table 35. Blue/Red IC3 Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
num_abms	integer1

Table 36. Blue/Red ABM Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
num_ints	integer2
num_awacs	integer1

Table 37. Blue/Red AD Site Relation

<i>Column Name</i>	<i>Format</i>
site_id *	integer4
adcc_oper	vchar(1)

Table 38. Blue/Red ADCC Site Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
lots_left	integer1

Table 39. Blue/Red Deploy Force Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
site_id *	integer4
num_lots	integer1

Table 40. Blue/Red Deploys Relation

<i>Column Name</i>	<i>Format</i>
pd_fm *	vchar(6)
next_fm	vchar(6)
prev_fm	vchar(6)

Table 41. PD Form Schedule Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
maxLots	integer1

Table 42. Blue/Red Force Relation

<i>Column Name</i>	<i>Format</i>
sys_id *	integer1
units_left	integer2
rtetgt	integer2

Table 43. Blue/Red Target Force Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(4)
unit_status	vchar(1)
location	integer4

Table 44. Blue/Red Units Relation

<i>Column Name</i>	<i>Format</i>
bmr_tail_nr *	vchar(5)
tnk_tail_nr *	vchar(5)

Table 45. Blue/Red Mated With Relation

<i>Column Name</i>	<i>Format</i>
snf_tailnr *	vchar(5)
sub_tailnr *	vchar(5)

Table 46. Blue/Red Stored On Relation

<i>Column Name</i>	<i>Format</i>
tailnr *	vchar(5)
opt_1	vchar(1)
opt_2	vchar(1)
opt_3	vchar(1)
opt_4	vchar(1)
entry_area	vchar(2)
exit_area	vchar(2)
rec_theater	vchar(5)
load_mode	vchar(1)
abmsleft	integer1
bombsleft	integer1
per_exec	integer1

Table 47. Blue/Red Bomber Vehicle Relation

<i>Column Name</i>	<i>Format</i>
tailnr *	vchar(5)
rte_seq *	integer1
tgt_seq *	integer1
tgt_site	integer4
wpn_type	vchar(1)

Table 48. Blue/Red Bomber Targets Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(5)
opt_1	vchar(1)
opt_2	vchar(1)
opt_3	vchar(1)
opt_4	vchar(1)
tgt_sect	integer1
rec_sect	integer1

Table 49. Blue/Red Fighter Vehicle Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(5)
nr *	integer1
tgt_site	integer4

Table 50. Blue/Red Fighter Bombs Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(5)
opt_1	vchar(1)
opt_2	vchar(1)
opt_3	vchar(1)
opt_4	vchar(1)
sector	integer1

Table 51. Blue/Red SNF Vehicle Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(5)
nr *	integer1
tgt_site	integer4

Table 52. Blue/Red SNF Warheads Relation

<i>Column Name</i>	<i>Format</i>
tail_nr *	vchar(5)
opt_1	vchar(1)
opt_2	vchar(1)
opt_3	vchar(1)
opt_4	vchar(1)
tgt_site	integer4

Table 53. Blue/Red NNF Vehicle Relation

Appendix C. IDEF₀ Diagrams

Simulation IDEF₀ Diagrams and Abstracts.

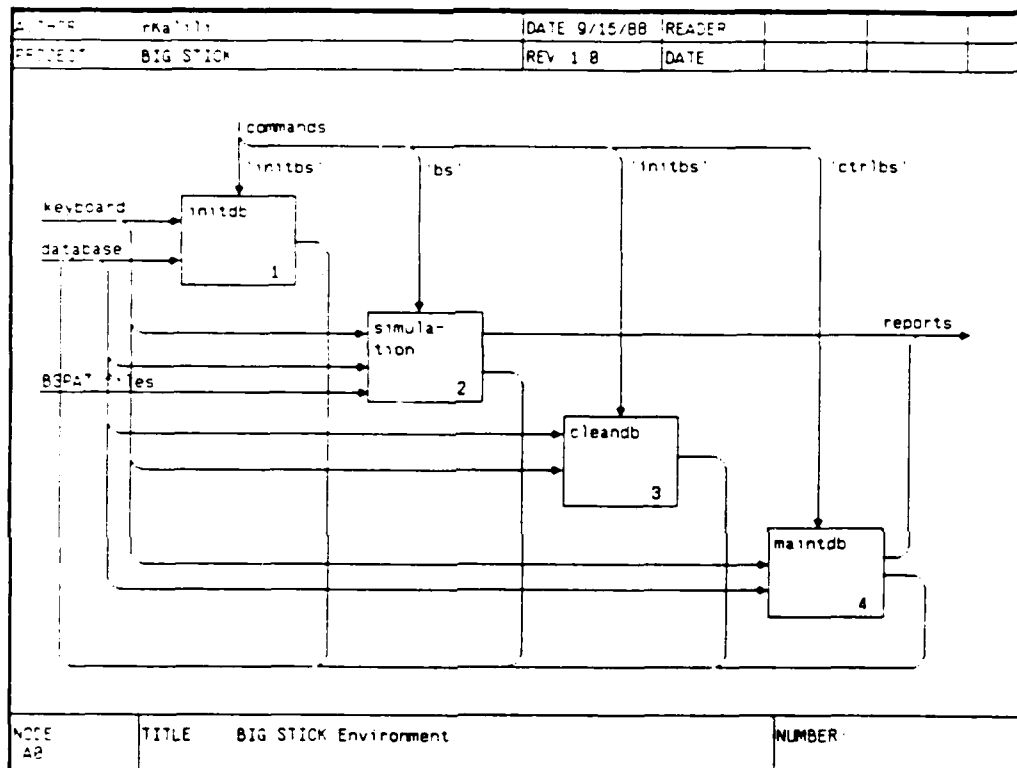


Figure 11. A0 - BIG STICK Environment IDEF₀ Diagram

Abstract. Overall BIG STICK Simulation Environment consists of four functions. Initdb prepares the database for execution of the simulation. Simulation controls the execution of the simulation by the students. Cleandb allows the controller to restore the database to an original state. Maintdb allows the controller to modify the database relations for maintenance and enhancement of the simulation.

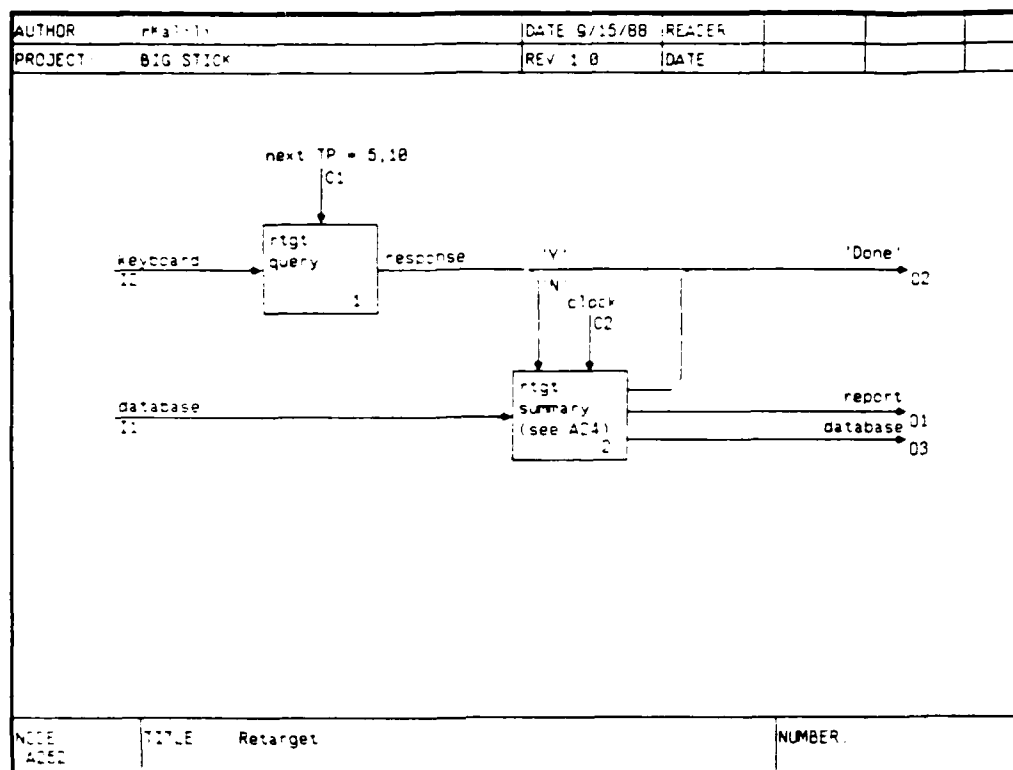


Figure 13. A252 - Retarget IDEF₀ Diagram

Abstract. Retargeting consists of a query and actual retargeting which is conducted in a manner very similar to the Force Targeting part of the simulation.

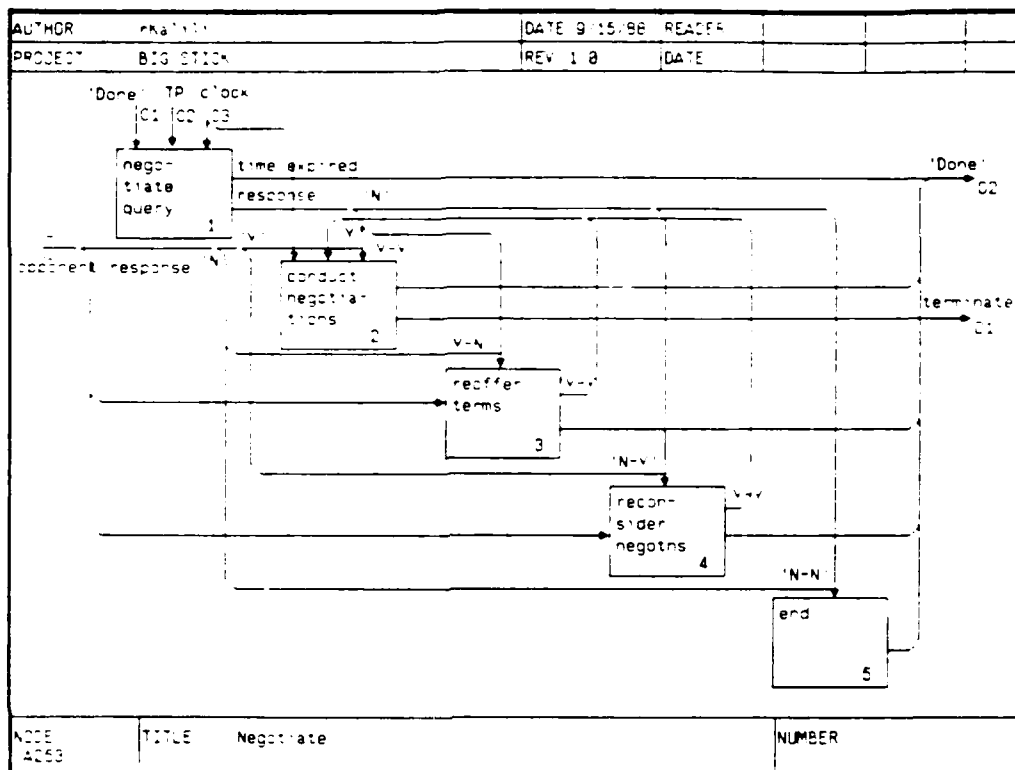


Figure 14. A253 - Negotiate IDEF₀ Diagram

Abstract. Negotiations results from specific responses from the student teams. This diagram captures the decision process required for negotiations to occur.

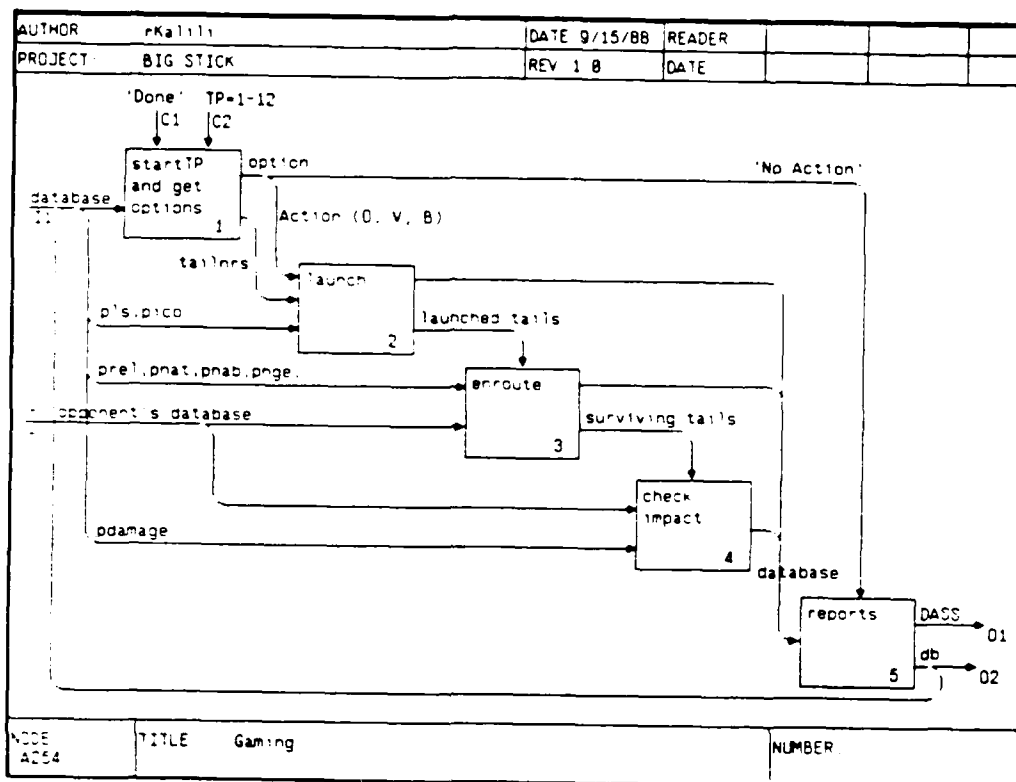


Figure 15. A254 - Gaming IDEF₀ Diagram

Abstract. Gaming is an iterative process through 5 phases for each force system that is executed by the student team. In general, one iteration creates one time period (TP).

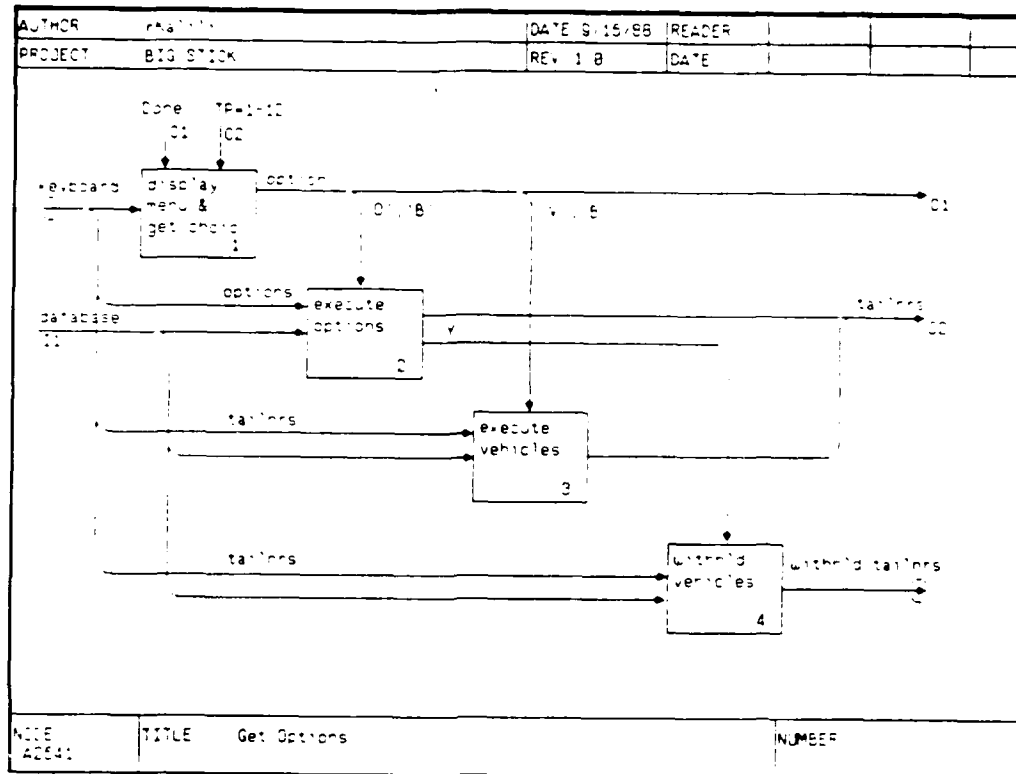


Figure 16. A2541 - Get Options IDEF₀ Diagram

Abstract. After the menu is displayed, the simulation will proceed to get the team's selection for options or vehicles to execute or both. Student teams may also withhold vehicles. Teams may also "do nothing" in which case the simulation will simply assess damage inflicted on the team's assets.

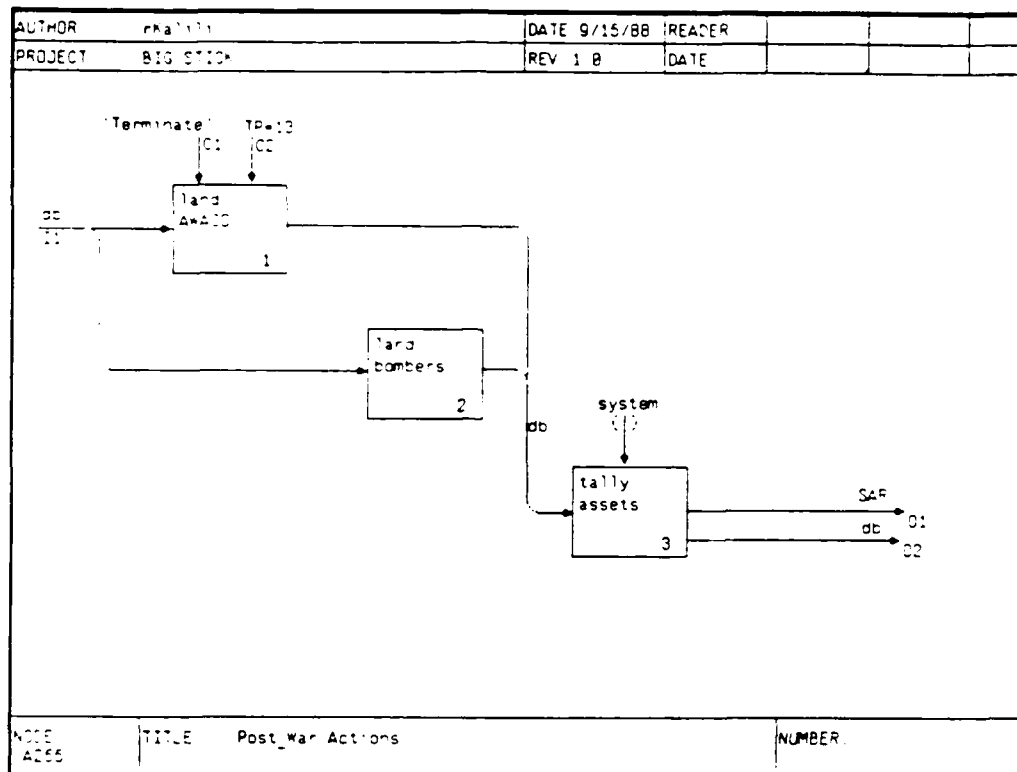


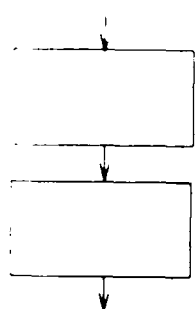
Figure 17. A255 - Post-War Actions IDEF₀ Diagram

Abstract. Post-War Actions occur at the end of the simulation. This involves landing the force systems that are airborne which do not normally land during gaming. Final scoring of the results are conducted.

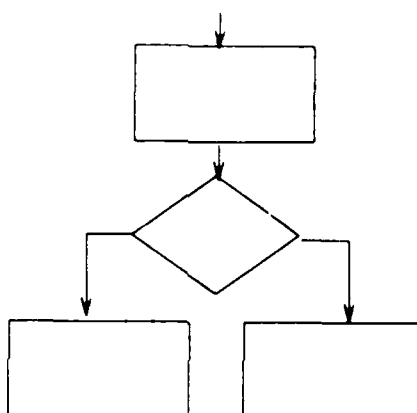
Appendix D. *Simulation Flowcharts*

The following flowcharts depict the sequence of activities required by the force employment portion of the simulation. These flowcharts provide detail to IDEF₀ diagram modules. Symbols used in the flowcharts are as follows.

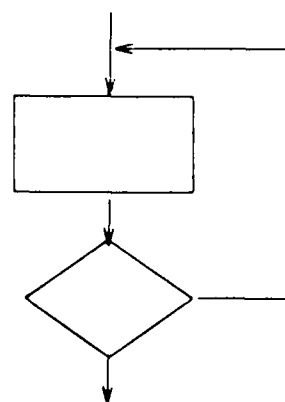
FLOWCHART SYMBOLS



Sequence



Decision



Repetition



Printout



External Interface



- Simple Connector or
Connector to IDEF Diagram



- Terminate

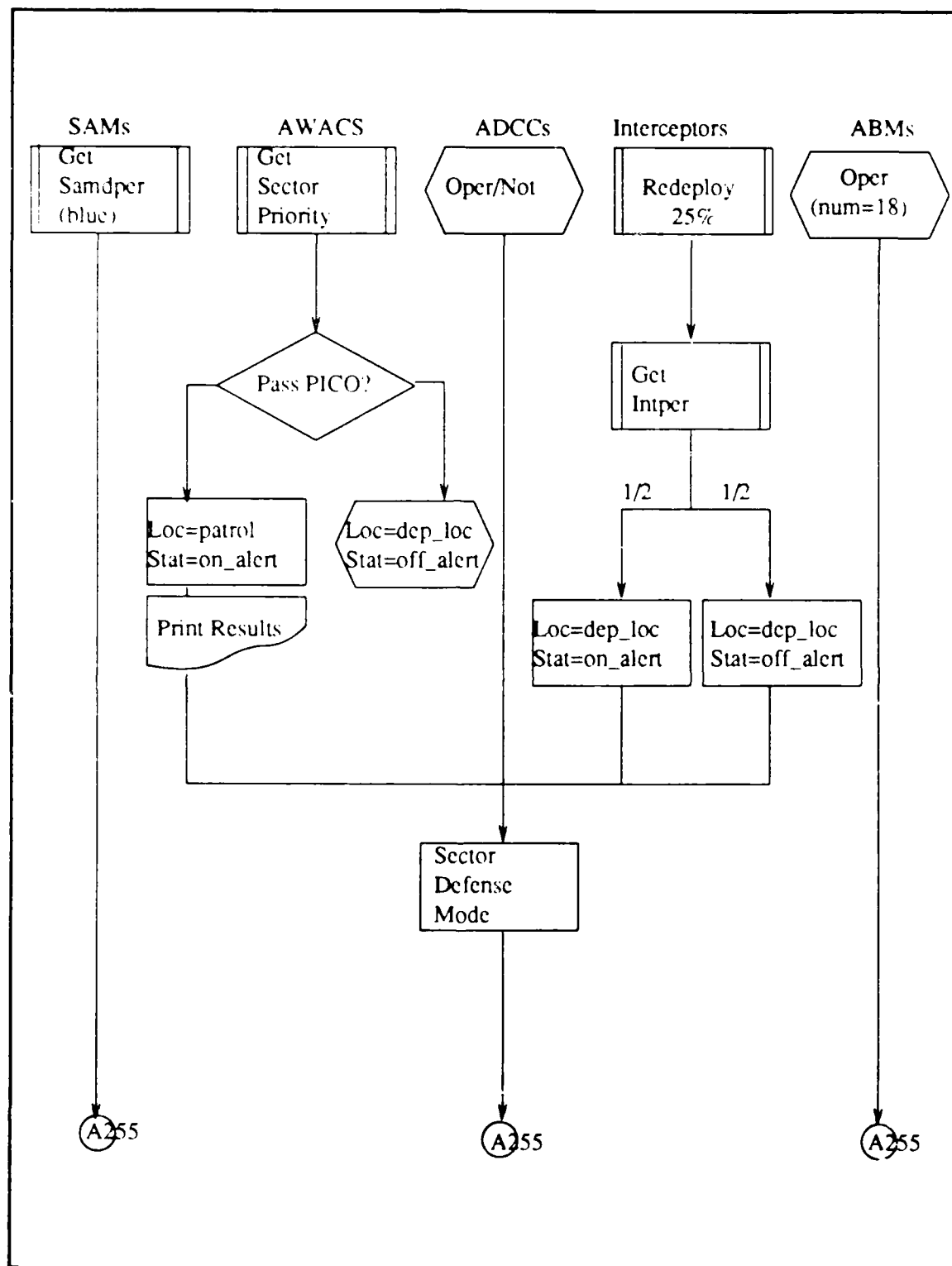


Figure 18. A251 - Defense Systems Pre-War Actions Flowchart

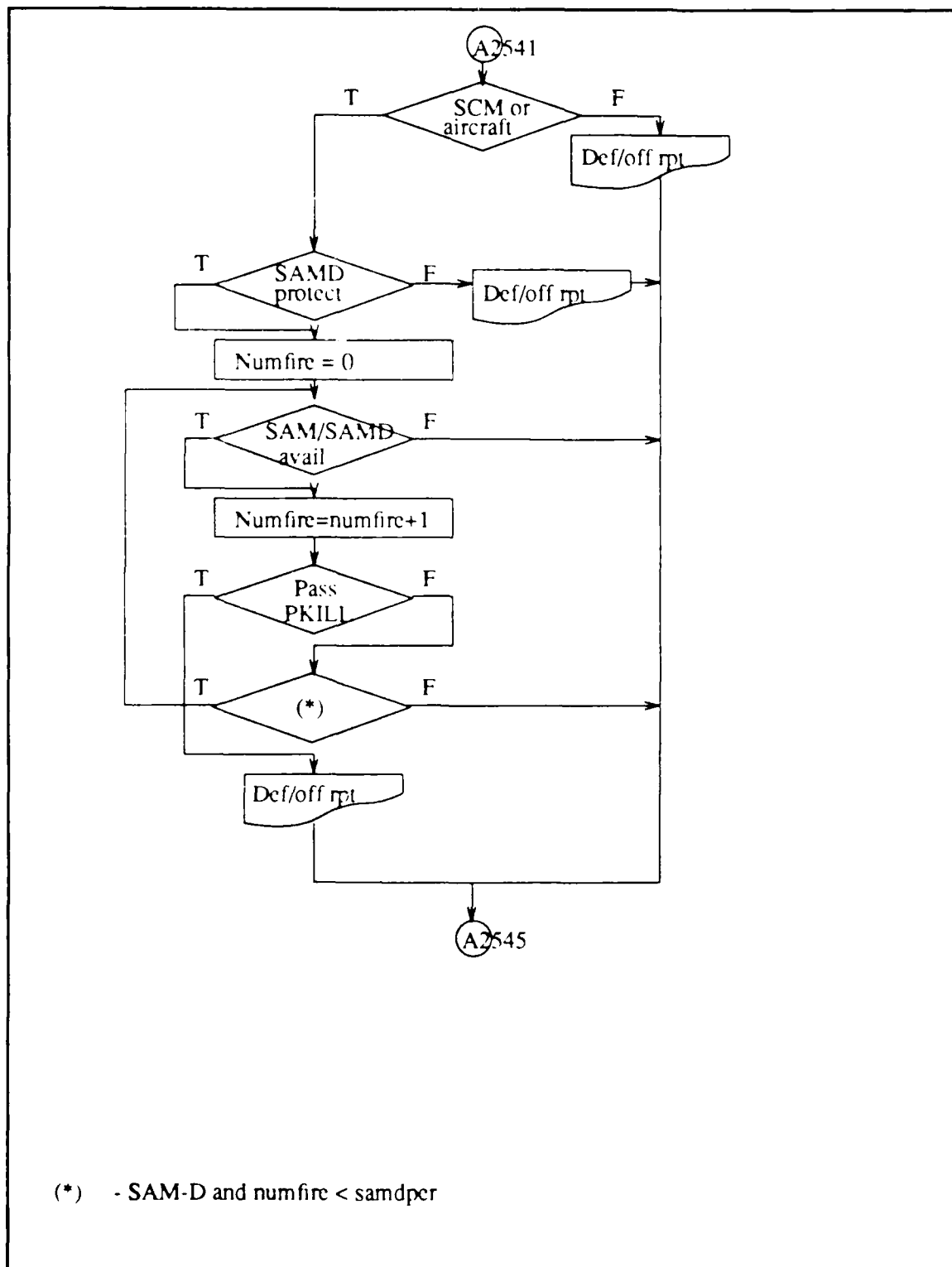


Figure 19. A2543/5 - SAM and SAM-D Enroute & Reports Flowchart

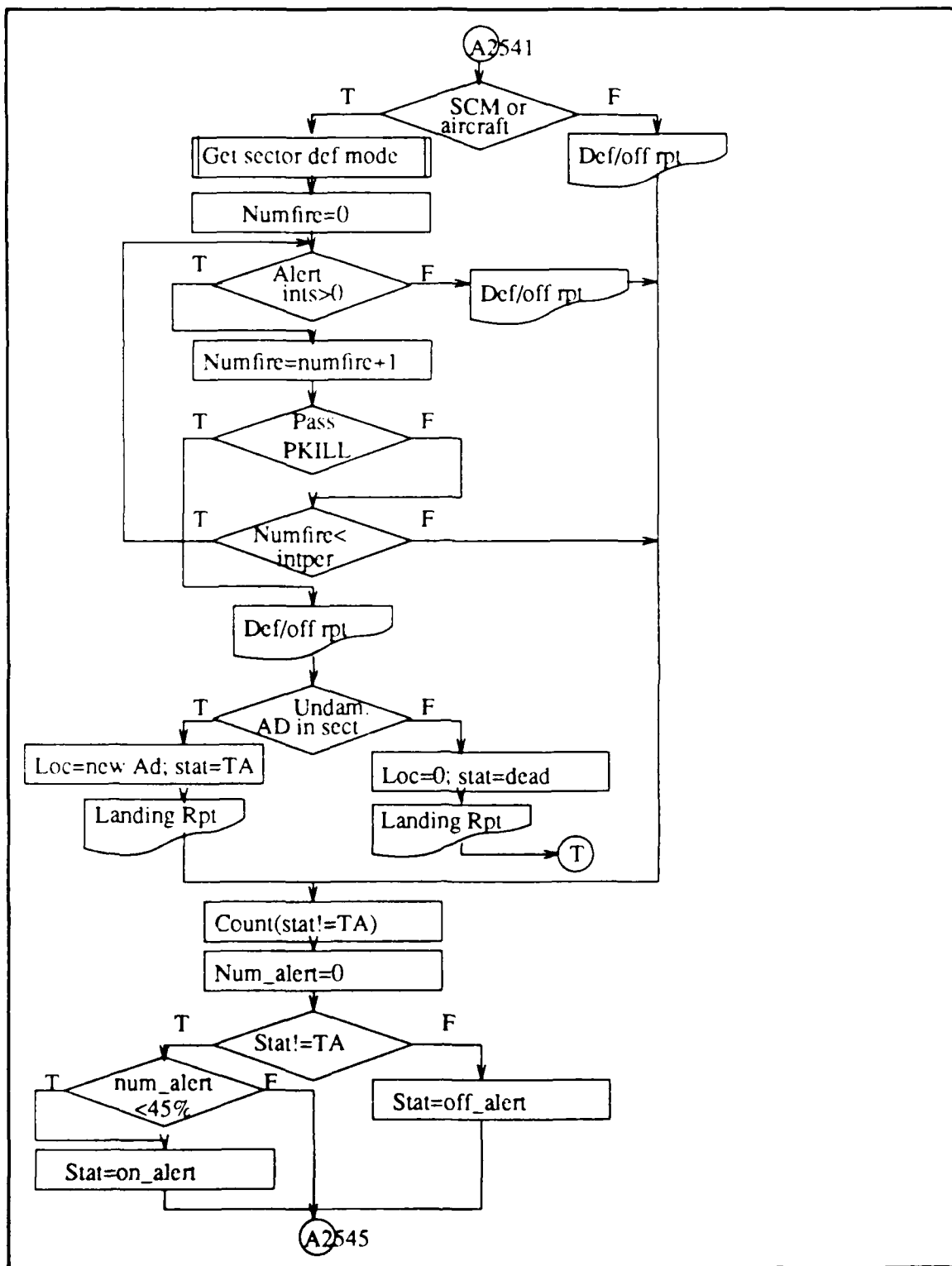


Figure 20. A2543/5 - Interceptor Enroute & Reports Flowchart

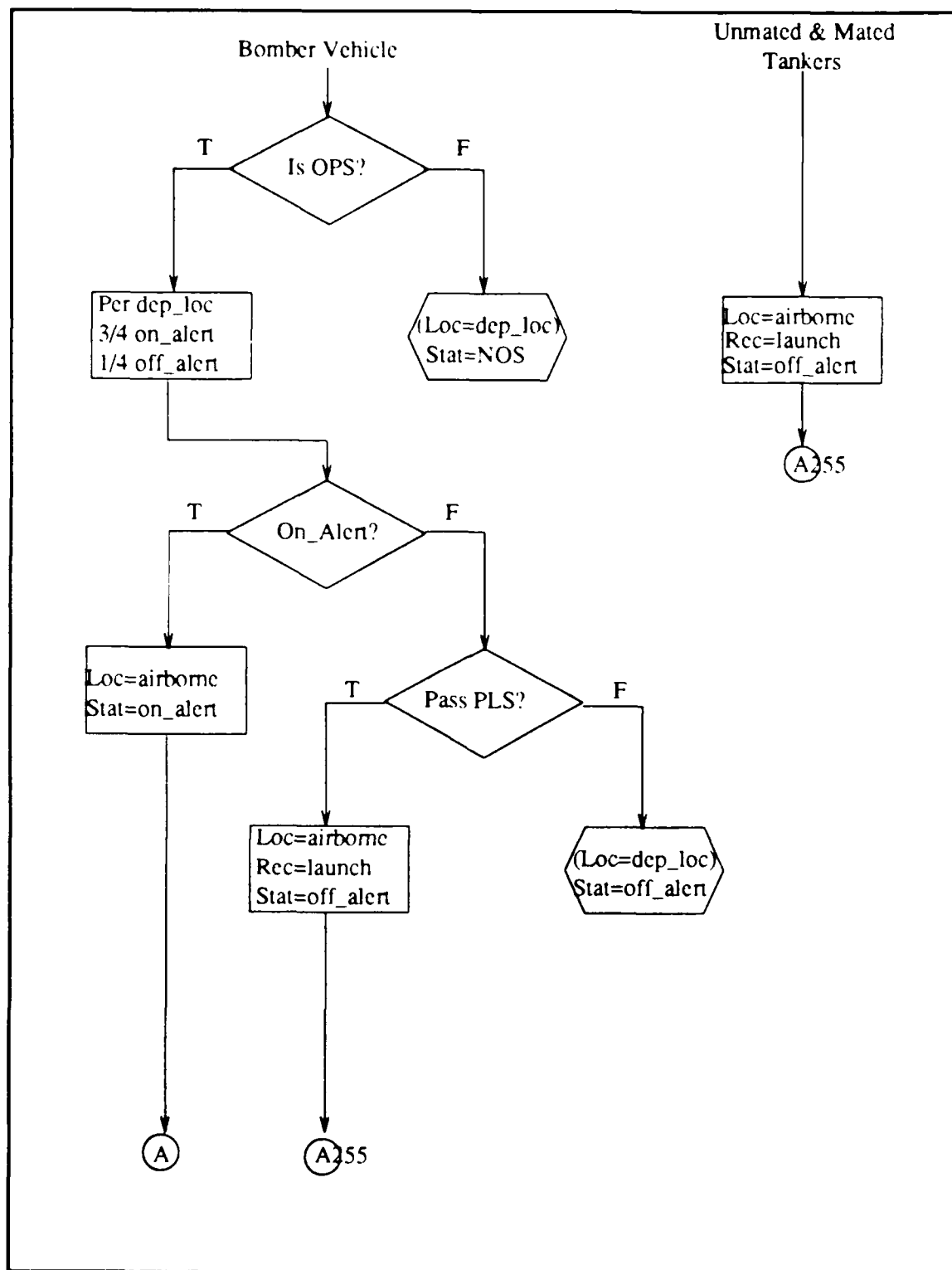


Figure 21. A251 & A2542 - Bomber Pre-War Actions & Launch Flowchart

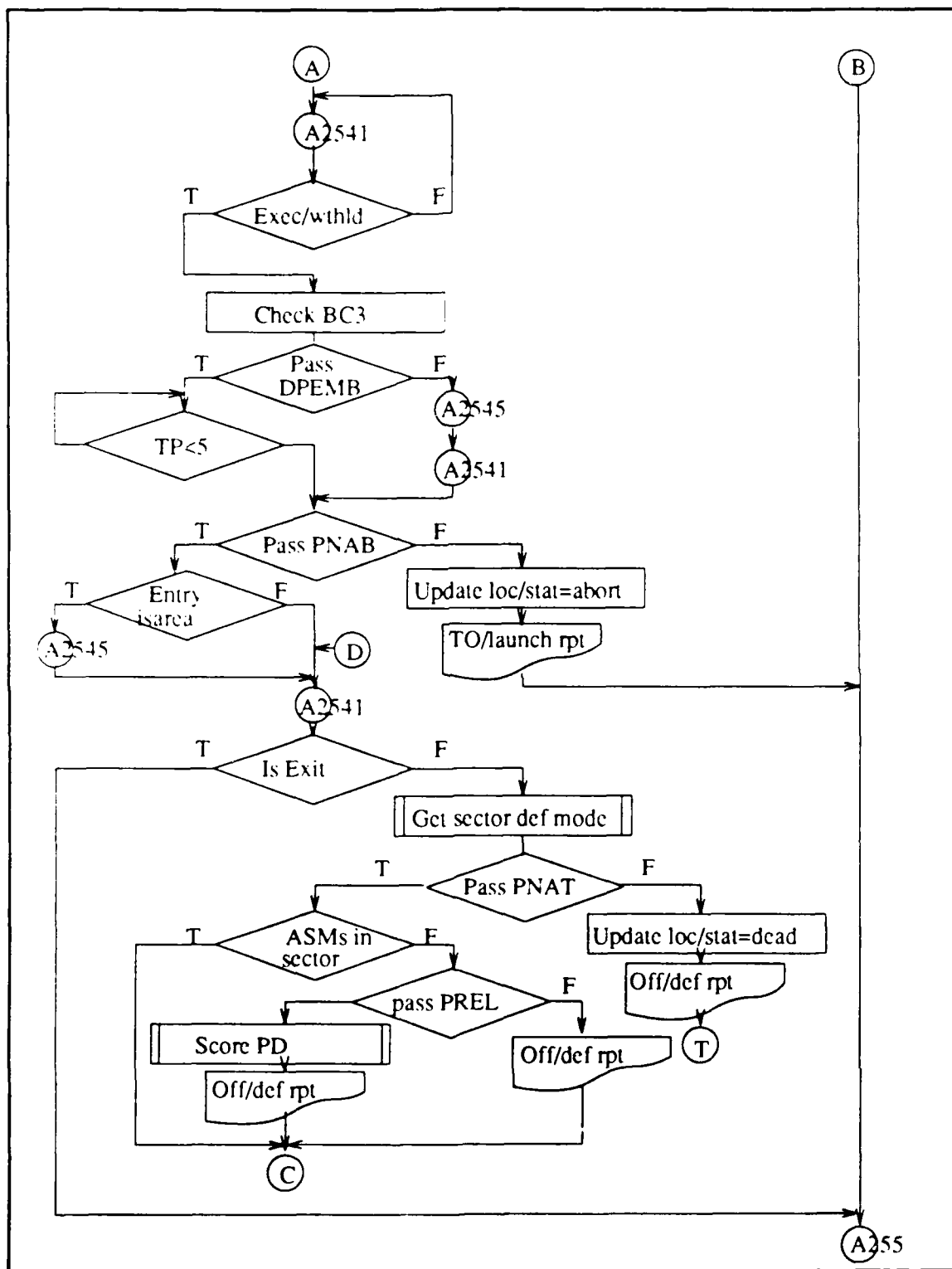


Figure 22. A2543/4/5 - Bomber Enroute, Impact, & Reports Flowchart

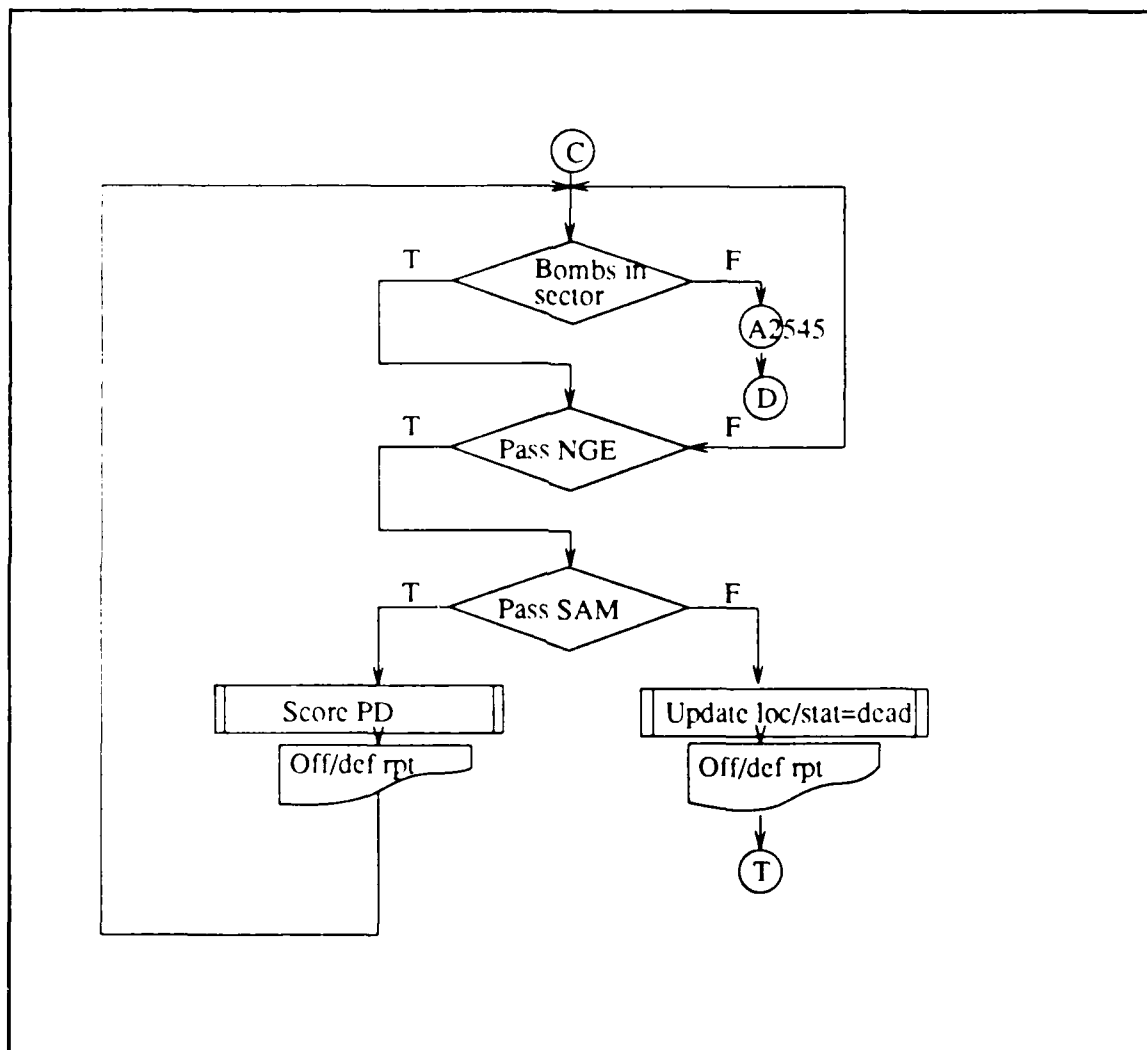


Figure 23. A2543/4/5 (cont) - Bomber Enroute, Impact, & Reports Flowchart

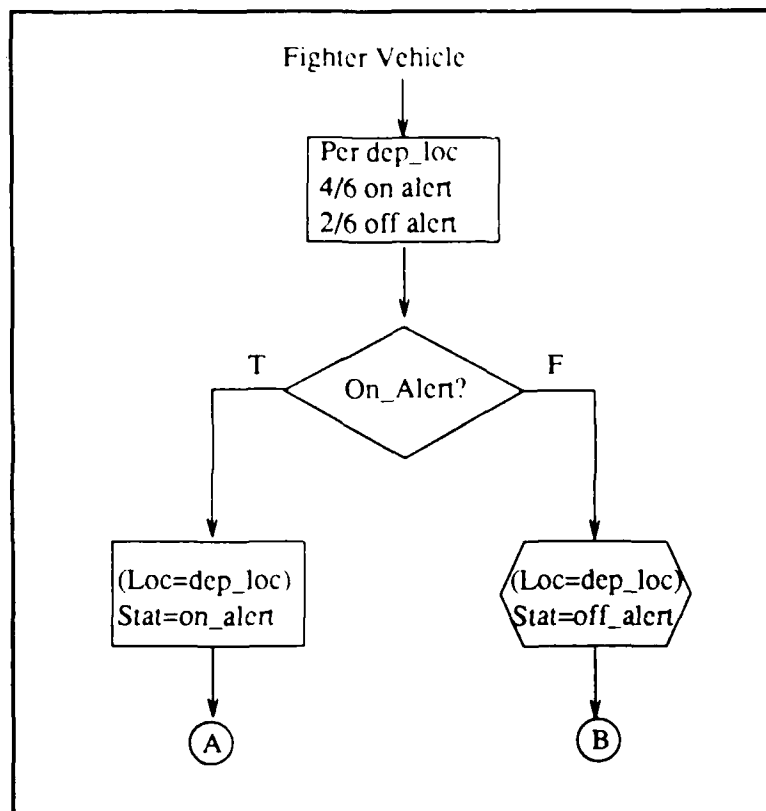


Figure 24. A251 - Fighter Pre-War Actions Flowchart

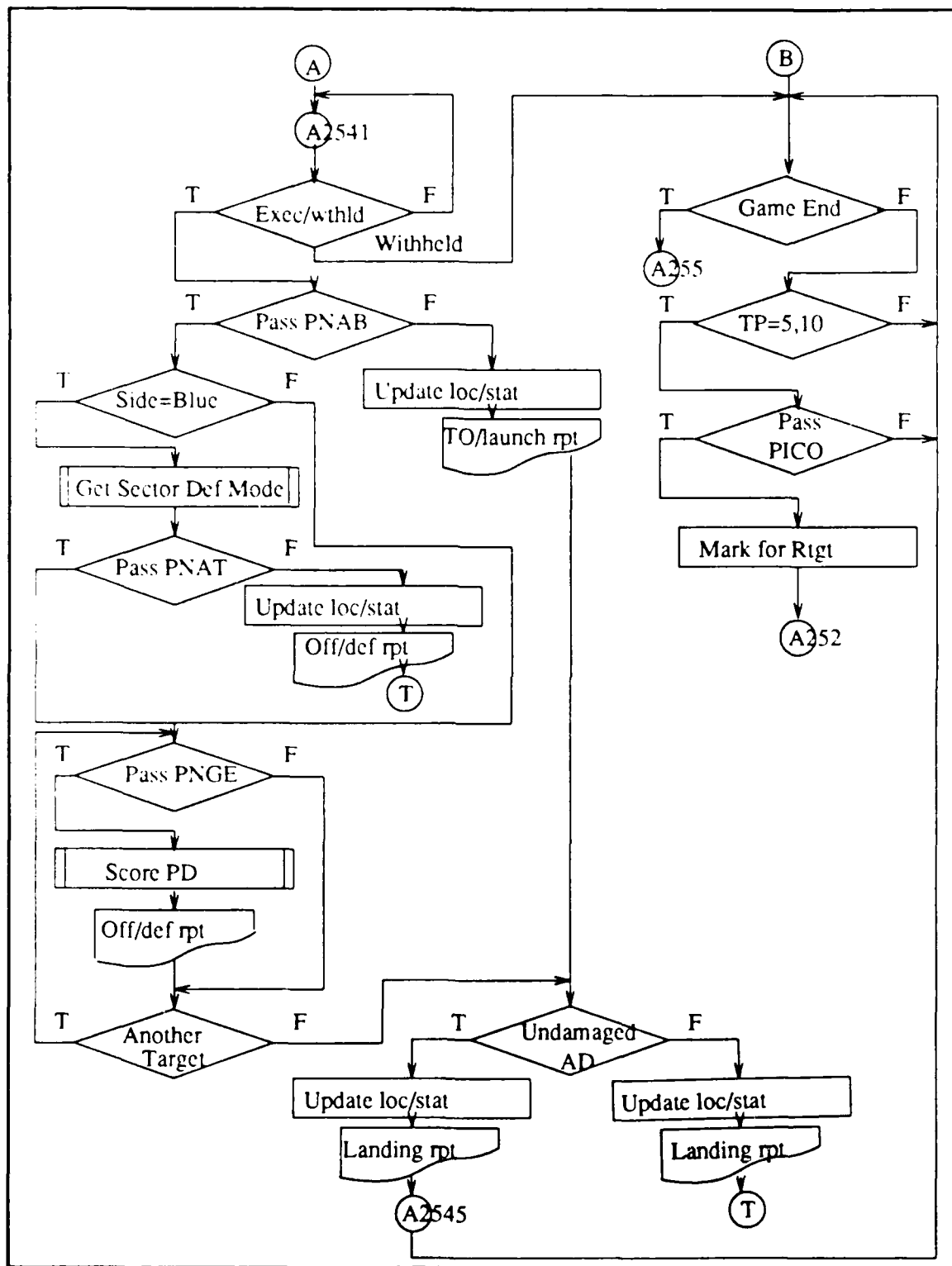


Figure 25. A2542/3/4/5 - Fighter Launch, Enroute, Impact, & Reports Flowchart

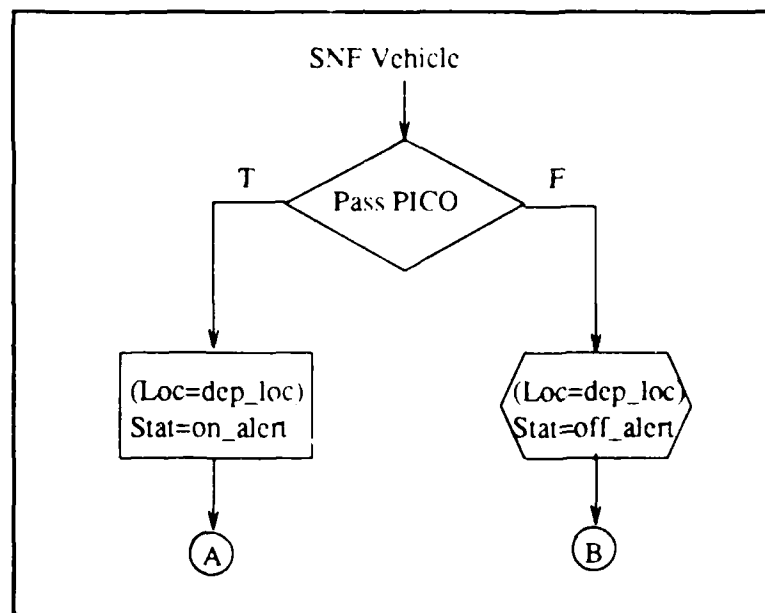


Figure 26. A251 - SNF Pre-War Actions Flowchart

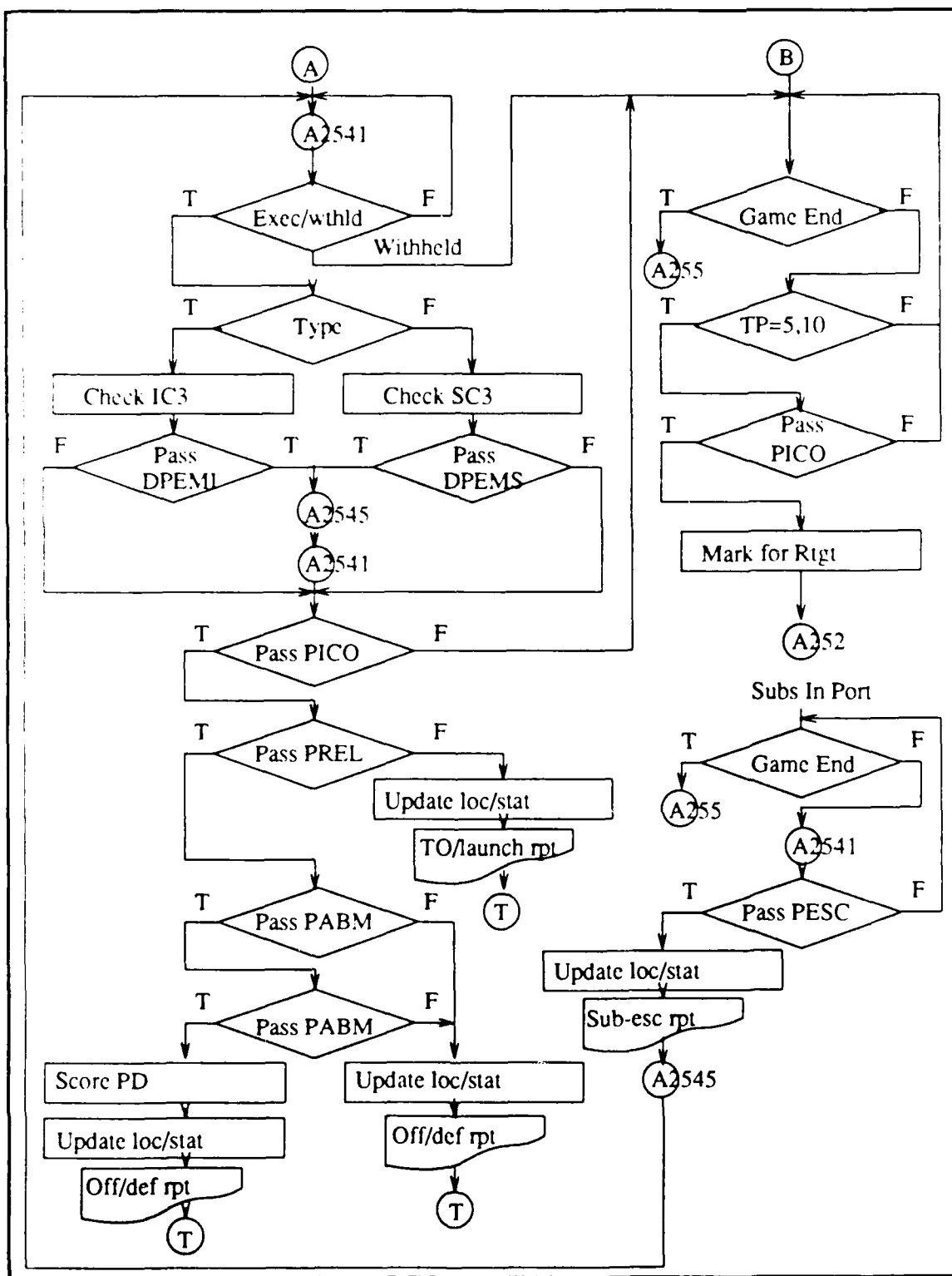


Figure 27. A2542/3/4/5 - SNF Launch, Enroute, Impact, & Reports Flowchart

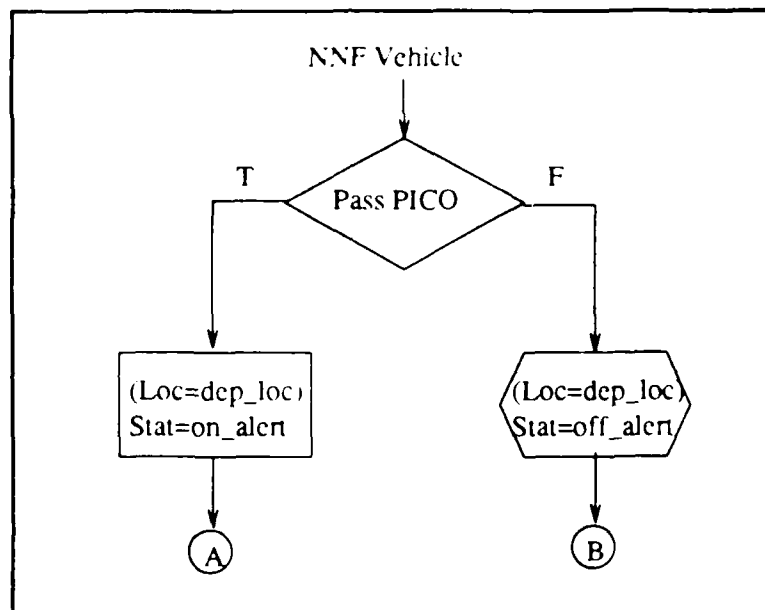


Figure 28. A251 - NNF Pre-War Actions Flowchart

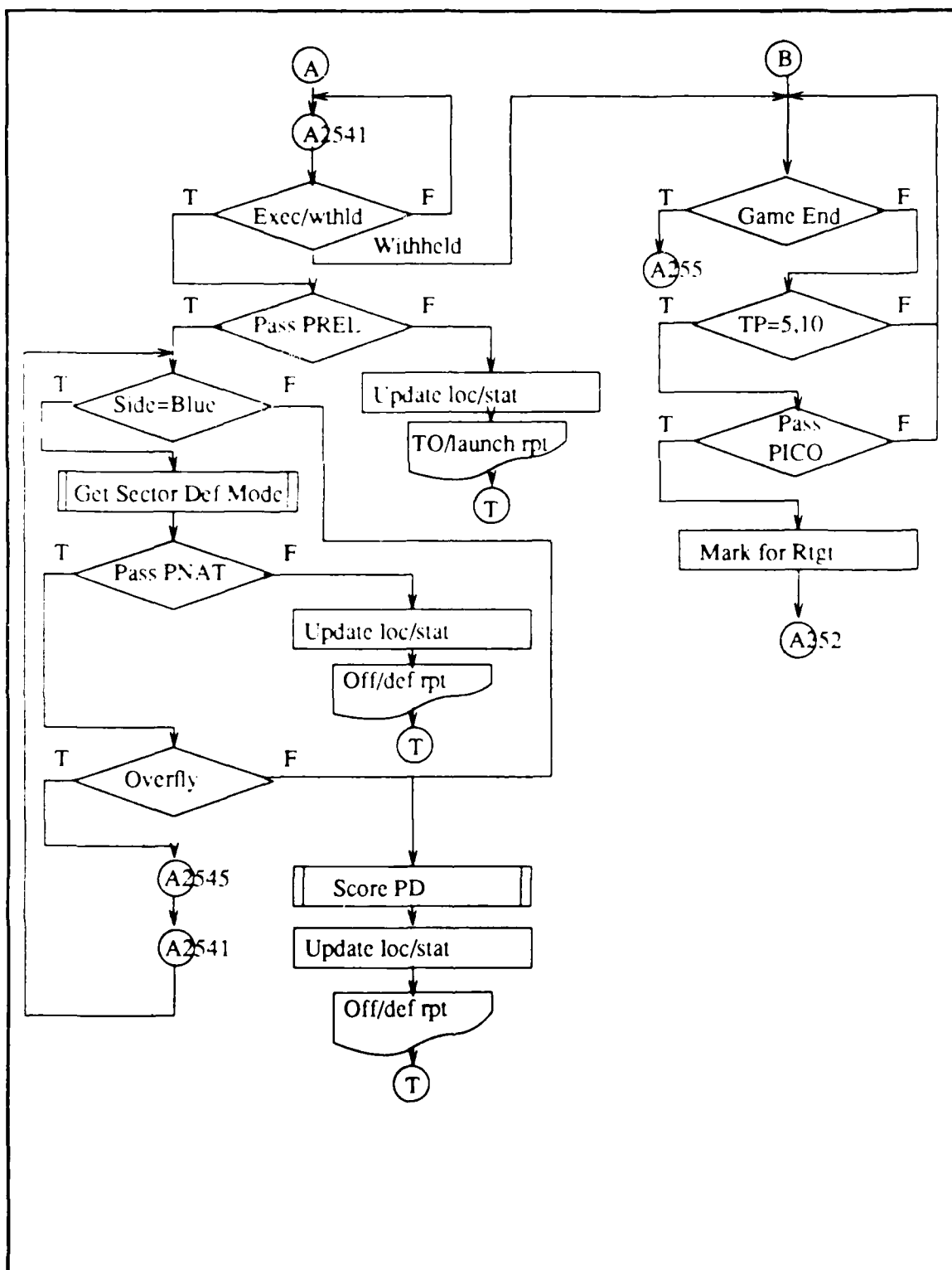


Figure 29. A2542/3/4/5 - NNF Launch, Enroute, Impact, & Reports Flowchart

Bibliography

1. *Automatic Data Processing Systems and Procedures, BIG STICK I Maintenance Manual*. HQ Air University, Maxwell AFB AL, 1984. ACDY Division OI 171-2, Vol III.
2. *BIG STICK Instructions and Planning Guidance*. Air Command and Staff College, Air University, Maxwell AFB AL, 1987. Unpublished Manual.
3. Barry W. Boehm. A Spiral Model of Software Development and Enhancement. *IEEE Computer*, 61-72, May 1988.
4. Robert C. Bonczek, Clyde W. Holsapple, and Andrew B. Winston. *Micro Database Management Practical Techniques for Application Development*. Academic Press, Orlando FL, 1984.
5. Grady Booch. *Software Engineering With ADA*. Benjamin/Cummings Publishing Company, Menlo Park CA, 1987.
6. Daniel J. Cronin. *Microcomputer Data Security, Issues and Strategies*. Prentice Hall Press, New York NY, 1986.
7. Lt Col Daniel B. Fox. *A Conceptual Design for a Model to Meet the War-Gaming Needs of the Major Commands of the United States Air Force*. Technical Report AU-ARI-84-8, Airpower Research Institute, Air University Press, Maxwell AFB AL, July 1985.
8. Wilbert O. Galitz. *Handbook of Screen Format Design*. QED Information Sciences, Wellesley MA, 1985.
9. Ian G. Gibbs. *Dictionary of Gaming, Modelling & Simulation*. Sage Publications, Beverly Hills CA, 1978.
10. G. R. Gladden. Stop the Life-Cycle, I Want to Get Off. *Software Engineering Notes*, 7:35-39, April 1982.
11. Francis P. Hoerber. *Military Applications of Modeling*. Gordon & Breach Science Publishers, New York NY, 1981.
12. Capt James R. Jansen. *Redesign of the Joint Planning Exercise (JPLAN)*. Master's thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1987. AFIT/GCS/ENG/87D-15.
13. Henry F. Korth and Abraham Silberschatz. *Database System Concepts*. McGraw-Hill Book Company, New York, 1986.
14. David M. Kroenke and Donald E. Nilson. *Database Processing For Microcomputers*. Science Research Associates, Chicago IL, 1986.
15. Capt Mark S. Kross. *Developing New User Interfaces for the Theater War Exercise*. Master's thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1987. AFIT/GCS/ENG/87D-19.
16. Daniel D. McCracken and Michael A. Jackson. Life Cycle Concept Considered Harmful. *Software Engineering Notes*, 7:29-32, April 1982.

17. Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill Book Company, New York, 1987.
18. M. E. Sime and M. J. Coombs. *Designing for Human-Computer Communication*. Academic Press, London, 1983.
19. Henry Simpson. *Design of User Friendly Programs for Small Computers*. McGraw-Hill Book Company, New York, 1985.
20. Lt Col Richard P. Smith. Computer Assisted Wargaming At The Air University. *1981 Winter Simulation Conference Proceedings, IEEE*, 679-684, 1981.
21. Major Charles P. Williams. *BIG STICK Planning and Analysis Tools (BSPAT)*. Technical Report 87-2745, Air Command and Staff College, Air University, Maxwell AFB AL, 1987.

Vita

Capt Ruth M. Kalili [REDACTED] Rev and Mrs John Kalili. She graduated from Kamehameha High School [REDACTED] She received a B.S. in computer science and mathematics from the University of Denver in 1984 and an M.B.A. with a concentration in contracting and acquisition from Western New England College in 1987. She previously served as a computer acquisition and cost analyst at the Air Force Computer Acquisition Center.

[REDACTED]
[REDACTED]

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			5. MONITORING ORGANIZATION REPORT NUMBER(S)	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GCS/ENG/88D-12			7a. NAME OF MONITORING ORGANIZATION	
NAME OF PERFORMING ORGANIZATION School of Engineering		6b. OFFICE SYMBOL (If applicable) AFIT/ENG	7b. ADDRESS (City, State, and ZIP Code)	
ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology Wright-Patterson AFB OH 45433-6583			9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Wargaming Center		8b. OFFICE SYMBOL (If applicable) AUCADRE/WG	10. SOURCE OF FUNDING NUMBERS	
9c. ADDRESS (City, State, and ZIP Code) Maxwell AFB AL 36112-5532			PROGRAM ELEMENT NO.	TASK NO.
			PROJECT NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) REDESIGN AND REHOST OF THE BIG STICK STRATEGIC NUCLEAR WARGAME SIMULATION (UNCLASSIFIED)				
12. PERSONAL AUTHOR(S) Ruth M. Kalili, Capt, USAF				
13a. TYPE OF REPORT MS Thesis		13b. TIME COVERED FROM TO	14. DATE OF REPORT (Year, Month, Day) 1988 December	15. PAGE COUNT 97
16. SUPPLEMENTARY NOTATION				
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP		
12	05		Database, Software Engineering, Simulation	
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Thesis Advisor: Mark A. Roth, Capt, USAF Assistant Professor of Electrical Engineering and Computer Science				
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Mark A. Roth, Capt, USAF			22b. TELEPHONE (Include Area Code) 513-255-3576	22c. OFFICE SYMBOL AFIT/ENG

Approved for release in
accordance with E.O. 13526
10 Jan. 1989

Item 19.

The strategic nuclear wargame BIG STICK is a two-sided, interactive, computer simulation used by the Air Command and Staff College to assist students in learning about real-world nuclear war planning.

Currently, the simulation is played on the Honeywell H6000 mainframe. Shortcomings of the simulation are the "user-hostile" environment, a rigid input format, the unforgiving and inflexible user interface, and the fixed file system that makes data changes and program enhancements difficult.

The purpose of this thesis effort was to redesign and rehost the BIG STICK simulation to the Zenith Z-158 classroom microcomputers.

Game sites, fixed and controlled force assets, expected value probabilities, and exercise constraints are now stored in a relational database designed using the entity-relationship (E-R) model.

The user interface was redesigned using general user-friendly program principles and guidelines to provide a screen oriented environment for students to enter force selection, deployment, targeting, and employment inputs. Reports reflecting the results of student inputs were designed and developed as part of the interface using the PC INGRES database management system.

The actual game play portion of the simulation was designed using top-down, hierarchical decomposition. Implementation of the design into program code is not included in the scope of this thesis.