NUREG/CR-6116
INEL-94/0039
Vol. 10

Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0

Data Loading Manual

Manuscript Completed: January 1995
Date Published: April 1995

Prepared by
R. L. VanHorn, L. M. Wolfram,
R. D. Fowler, S. T. Beck, C. L. Smith

Idaho National Engineering Laboratory
Managed by the U.S. Department of Energy

Lockheed Idaho Technologies Company
Idaho Falls, ID 83415

Prepared for
Division of Systems Technology
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001
NRC Job Code W6241

DISCLAIMER

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

ABSTRACT

The Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) suite of programs can be used to organize and standardize in an electronic format information from probabilistic risk assessments or individual plant examinations. The Models and Results Database (MAR-D) program of the SAPHIRE suite serves as the repository for probabilistic risk assessment and individual plant examination data and information. This report demonstrates by examples the common electronic and manual methods used to load these types of data. It is not a stand alone document but references documents that contribute information relative to the data loading process. This document provides a more detailed discussion and instructions for using SAPHIRE 5.0 only when enough information on a specific topic is not provided by another available source.

W6241—Plant Database Development for SAPHIRE

Documents in NUREG/CR-6116 Report, Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0

Volume 1 - Technical Reference Manual

Volume 2 - Integrated Reliability and Risk Analysis System (IRRAS) Reference Manual

Volume 3 - Integrated Reliability and Risk Analysis System (IRRAS) Tutorial Manual

Volume 4 - Systems Analysis and Risk Assessment (SARA) System Reference Manual

Volume 5 - Systems Analysis and Risk Assessment (SARA) System Tutorial Manual

Volume 6 - Graphical Evaluation Module (GEM) Reference Manual

Volume 7 - Fault Tree, Event Tree, and Piping & Instrumentation Diagram (FEP) Editors Reference Manual

Volume 8 - Models and Results Database (MAR-D) Reference Manual

Volume 9 - Validation and Verification (V&V) Manual

Volume 10 - Data Loading Manual

Previous Reports in the Series

R. D. Fowler, et al., *Procedures Guide for Extracting and Loading Probabilistic Risk Assessment Data into MAR-D using IRRAS 2.5*, NUREG/CR-5520, EGG-2630, November, 1991.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

The Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) refers to a set of several microcomputer programs that were developed to create and analyze probablistic risk assessments (PRAs), primarily for nuclear power plants (NPPs). The programs currently in this set include: MAR-D (Models and Results Database—used primarily for data loading), IRRAS (Integrated Reliability and Risk Analysis System—used primarily for the purpose of performing those functions necessary to create and analyze a complete PRA), SARA (Systems Analysis and Risk Assessment—used primarily to perform limited sensitivity analyses on a loaded PRA), and FEP (Fault tree, Event tree, and Piping and Instrumentation graphic editor—graphical editors). A summary of the four programs that currently comprise SAPHIRE is given in the Foreword. Each of these programs performs a specific function in developing a PRA from the conceptual state all the way to publication. This manual demonstrates how to load an existing database for use by the analysis programs, primarily IRRAS and SARA.

The database loading processing depends strongly on MAR-D, whose primary function is to create a data repository for completed PRAs and Individual Plant Examinations (IPEs) by providing input, conversion, and output capabilities for data used by the analysis programs. MAR-D provides facilities to load, store, and extract all of the information associated with a PRA. This information includes the following: event trees (both logic and graphics); sequences (both logic and cut sets); fault trees (logic, graphics, and cut sets); basic event failure rates and descriptions; plant damage states, basic event attributes, failure modes, system descriptions, component descriptions, seismic data, location descriptions, class attributes, and family information. Some of the elements of MAR-D are in the IRRAS and SARA software to allow these programs to load and unload data in the MAR-D format.

As PRAs and IPEs are submitted to the NRC for review, MAR-D can be used to convert the models and results from the study for use with the analysis programs. Additionally, "flat" or ASCII files can be created using IRRAS or a text editor, or a combination of both. IRRAS can also be used to extract files from or load files into MAR-D to help the user set up a complete PRA database. This manual presents a sample database and discusses the processes available to load the data described above.

FOREWORD

The U.S. Nuclear Regulatory Commission has developed a powerful suite of personal computer programs for the performance of probabilistic risk assessments (PRAs). This suite of programs, known as the Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE), allows an analyst to perform many of the functions necessary to create, quantify, and evaluate the risk associated with a facility or process being analyzed. These programs include software to define the database structure, to create, analyze, and quantify the data, and to display results and perform sensitivity analyses. The programs included in this suite are as follows: Models And Results Database (MAR-D) software, Integrated Reliability and Risk Analysis System (IRRAS) software, Systems Analysis and Risk Assessment (SARA) software, and Fault tree, Event tree, and P&ID (FEP) graphical editor software. Each of these programs performs a specific function in taking a PRA from the conceptual state all the way to publication.

MAR-D is a program that is used primarily for PRA data loading. This program defines a common relational database structure that is used by the entire suite of programs. This structure allows all of the software to access and manipulate data created by other software in the system without performing a lengthy conversion. Therefore, data created by IRRAS is immediately available to SARA for sensitivity analysis. The MAR-D program also provides the facilities for loading and unloading of PRA data from the relational database structure used to store the data. A simple ASCII data format is used for interchange with other PRA software not included in NRC's suite of programs. This feature allows compatibility with previously developed software systems and allows for maximum data interchange. Elements of this software are included with both IRRAS and SARA to allow these programs to load and unload data in the MAR-D format. Normally, the entire MAR-D software is used only by those performing a data loading function and is not required by the end user. Documentation for MAR-D Version 5.0 is available as NUREG/CR-6116, Volume 8. It should be noted that whenever the MAR-D database structure is changed, it necessitates changes in the remaining codes (i.e., IRRAS, SARA, and FEP). Therefore, the code version numbers are changed in unison. Each version set must be used together to maintain compatibility.

IRRAS is a program developed for the purpose of performing those functions necessary to create and analyze a complete PRA. This program includes functions to allow the user to create event trees and fault trees, to define accident sequences and basic event failure data, to solve system and accident sequence fault trees, to quantify cut sets, and to perform uncertainty analysis on the results. Also included in this program are features to allow the analyst to generate reports and displays that can be used to document the results of an analysis. Because this software is a very detailed technical tool, the user of this program should be familiar with PRA concepts and the methods used to perform these analyses. Although IRRAS has been designed to be user friendly and makes the process of performing a PRA easier, the complexity of this type of analysis requires a user with a more detailed understanding of PRA concepts than is required by other tools in this suite. The IRRAS 5.0 reference manual and tutorial are available as NUREG/CR-6116, Volumes 2 and 3, respectively. In addition, a technical document that provides information on the principles and algorithms used in the construction and operation of IRRAS and SARA is available as NUREG/CR-6116, Volume 1.

GEM provides a specific user interface tool for performing nuclear power plant operational event assessment using the SAPHIRE-based Accident Sequence Precursor (ASP) models and methods. GEM streamlines and automates select inputs and processes that are used to calculate conditional core damage probabilities (CCDPs) and provides outputs and reports consistent with ASP applications. With this program, a user can establish an analysis work area, make modifications to the model to represent the conditions of the operational event, and reprocess the models to automatically calculate the CCDP. This also includes the application or the appropriate recovery factors for the initiating event. For pressurized water reactors (PWRs), GEM also revaluates the probability of a reactor coolant pump seal loss of coolant accident (LOCA).

SARA is a program that allows the user to review the results of a PRA and to perform limited sensitivity analyses on these results. It is limited primarily to the extent that changes in the plant model can be accommodated by using the cut set editor. If other than simple changes are being simulated, then IRRAS should be used so that new cut sets can be accurately generated. This tool is intended to be used by a less technically-oriented user and does not require the level of understanding of PRA concepts required by IRRAS. With this program a user can review the information generated by a PRA analyst and compare the results to those generated by making limited modifications to the data in the PRA. Also included in this program is the ability to graphically display the information stored in the MAR-D database. This information includes event trees, fault trees, P&Ids and uncertainty distributions. The user of this program can gain a better understanding of the results of a PRA without getting into the details of the construction and analysis work behind the PRA. The SARA reference manual and tutorial are available as NUREG/CR-6116, Volumes 4 and 5, respectively.

FEP is a program developed to provide a common access to the suite of graphical tools developed for performing risk assessment. These tools include the graphical fault tree, event tree, and P&ID editors. The fault tree and event tree editors are available through IRRAS; however, the P&ID editor is only accessible through FEP. The fault tree editor allows the user to construct and modify graphical fault trees. The event tree editor allows the analyst to construct and modify graphical event trees. The P&ID editor allows the user to construct and modify plant drawings. These drawings can then be used to document the modeling used in a PRA. These editors are an integral part of a PRA. With the FEP tool, the user need not be concerned with the complexity of the IRRAS program if the need is only to generate one of these graphical displays. The FEP Reference Manual is available as NUREG/CR-6116, Volume 7.

ACKNOWLEDGMENTS

The authors would like to express their appreciation to Richard C. Robinson, the U.S. Nuclear Regulatory Commission (USNRC) Technical Monitor, for support and guidance in the database development for SAPHIRE project and to Tami A. Thatcher, Dawnie L. Judd, David R. Pack, and Larry D. Hilton for their significant contribution to the project.

ACRONYMS

CDF	core damage frequency
FEP	Fault Tree, Event Tree, and Piping and Instrumentation Diagram Editor code
FRANTIC	Formal Reliability Analysis, including Normal Testing Inspection and Checking code
INEL	Idaho National Engineering Laboratory
IPE	individual plant examination
IRRAS	Integrated Reliability and Risk Analysis System code
MAR-D	Models and Results Database code
NRC	Nuclear Regulatory Commission
PC	personal computer
P&ID	piping and instrumentation diagram
PRA	probabilistic risk assessment
SAPHIRE	Systems Analysis Programs for Hands-on Integrated Reliability Evaluations
SARA	Systems Analysis and Risk Assessment code
SETS	Set Equation Transformation System code
TM	test and maintenance

Systems Analysis Program for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0 Volume 10—Data Loading Manual

1. INTRODUCTION

This manual is designed to guide the user through the basic procedures necessary to enter probabilistic risk assessment (PRA) data into the Models and Results Database (MAR-D) using the Systems Analysis Programs for Hands-on Integrated Reliability Evaluation (SAPHIRE) suite of programs. A simple sample database presented in Section 3 demonstrates the data loading process. Where applicable, the discussion includes how the processes for loading the sample database relate to the actual processes used to load a larger PRA or individual plant examination (IPE) database.

The procedures in the manual were developed for use with SAPHIRE, Version 5.0, and Models and Results Database (MAR-D) Version 5.0 and may not apply to past or future versions. The manual provides guidance for efficient and accurate data entry. It is not intended to stand alone but to supplement existing documents. Therefore, the manual references the MAR-D Reference Manual,[1] the Integrated Reliability and Risk Analysis System (IRRAS) Reference Manual,[2] the IRRAS Tutorial Manual,[3] and the SAPHIRE Technical Reference Manual[4] as sources of additional information.

1.1 Loading PRA Results Program Overview

There is a continual need for nuclear plant risk information documented in PRAs and IPEs. The Idaho National Engineering Laboratory (INEL) is under contract with the U.S. Nuclear Regulatory Commission (NRC) to collect and load data and information for internally generated events from PRAs and IPEs into personal computer (PC) based databases. The NRC programs and projects for which these databases are useful include (a) prioritization, evaluation, and resolution of generic safety issues, (b) risk monitoring of plants, (c) assessment of operational events (i.e., event analysis), (d) evaluation of changes to technical specifications, and (e) evaluation of designs for concept plants.

The databases are being developed using a sophisticated PC-based program, which is also being produced by the INEL under contract with the NRC. This program, named SAPHIRE, is actually a suite of four individual programs:

1. Integrated Reliability and Risk Analysis System (IRRAS)

2. Systems Analysis and Risk Assessment (SARA)

3. Models and Results Database (MAR-D)

4. Fault Tree, Event Tree, and Piping and Instrumentation Diagram Editors (FEP).

The MAR-D system serves as the repository for PRA or IPE data and information contained in the database. It provides load routines for data in the correct format and output routines that will format data for use with other PRA tools. IRRAS is an integrated software tool that provides the ability to create and analyze event tree models, fault tree models, and accident sequences. SARA provides a means to change basic event failure rates and system models in the database for use in sensitivity analyses. The effects of such changes can then be evaluated through the calculation of the resultant core damage frequency (CDF), accident sequence probabilities, and importance measures. The results can be output in both a text and graphics format. FEP editors provide the capability to graphically build or edit event trees, fault trees, and piping and instrumentation diagrams (P&IDs). All SAPHIRE software tools have basically the same framework and are designed to operate from the same database.

Some electronic data generated by other computer applications can also be directly loaded (in ASCII format) into the SAPHIRE 5.0 software. This includes data from the

- Set Equation Transformation System (SETS).[5]

Output from SAPHIRE includes data that are compatible with the following software:

- Set Equation Transformation System (SETS)[5]
- Formal Reliability Analysis, including Normal Testing Inspection and Checking (FRANTIC).[6]

1.2 Assumptions and Recommendations

It is assumed that the SAPHIRE software has been loaded as described in Appendix A of the IRRAS Reference Manual.[2] This data loading manual predominately discusses the IRRAS program of the SAPHIRE suite. All of the processes discussed in this manual can be accomplished with IRRAS. The other programs in the suite apply more to other uses, though there is some overlap in abilities. It is assumed that the user is knowledgeable in the use of IRRAS. It is also assumed that the user has a basic level of knowledge concerning the use of event trees and fault trees in a PRA.

It is recommend that the user read Section 2.0 of the IRRAS Reference Manual prior to reading this manual.[2] Section 2.0 provides an overview of IRRAS with discussions concerning how to get around in the program menus, the IRRAS modules, and IRRAS database concepts. These concepts will be discussed only briefly in Section 2.

2. OVERVIEW OF DATABASE CONCEPTS

2.1 SAPHIRE Database Unit—The Family

The SAPHIRE analysis structure is divided into families. Since access to any SAPHIRE interactive database is obtained through the appropriate family, a family is the first thing that must be created. A family is any logical grouping of fault trees and event trees with their associated basic events, cut sets, reliability data, and descriptions. When a database family is created in one of the SAPHIRE suite of programs, a corresponding DOS subdirectory contained on the SAF50 subdirectory is also created (this assumes that SAPHIRE was installed in the SAF50 directory). After creation of the family, it is also necessary to *select* this family to allow input of data. The procedures for adding and selecting a family in IRRAS are shown in Appendix A of this report.

2.2 SAPHIRE Modules

SAPHIRE is structured so that the various functions are contained in individual modules or program units. These modules will be referred to frequently throughout this manual. The modules used predominantly in data loading are

- SELECT Family
- BUILD Fault Trees
- CREATE Event Trees
- MODIFY Data Base
- UTILITY Options.

These modules appear to a certain extent in all of the programs in the SAPHIRE suite. The IRRAS program is the only one that contains all of the modules and is the most commonly used in the data loading process. Additional detailed information on these and other modules can be found in both the IRRAS Reference Manual[2] and Tutorial.[3]

2.3 Base Versus Current Case Concepts

The interactive database in SAPHIRE consists of both a current (or working case) and a base case. These two analysis cases are not necessarily identical. When working in IRRAS, particularly in the report and utilities modules, it is a common query whether to use the base or current case for a particular activity. This concept is very important when dealing with cut set generation and quantification for both fault trees and event trees. It is possible to have two sets of values for basic events, cut sets, and importance. This option allows the user to maintain a base case database that can be transferred to the current case (via the Generate Data option). Once in the current case, the data can be changed as necessary for analysis, without losing the base case database values and results.

All basic event data entered into the IRRAS Modify Data Base module is automatically placed in the base case database. When loading values, SAPHIRE will allow information to be input to the base case and/or the current case database. Unless queried during the process, any analysis performed uses the SAPHIRE program defaults to values and/or cutsets drawn from the current case database. Note that basic fault tree and event tree logic remains the same for both cases.

Note: ALTERNATE CASE = CURRENT CASE. The IRRAS Reference Manual[2] and some menus in both IRRAS and SARA refer to an ALTERNATE case. These terms should be considered synonymous.

2.4 File Management

There are several types of external files important to SAPHIRE for storing and accessing database information. All files associated with a particular database are stored in the subdirectory representing the family. The family subdirectory is found in the SAF50 subdirectory, as discussed in Section 2.1.

The external relation files reside in the family DOS subdirectory and maintain the permanent SAPHIRE interactive database. The types of relations data include family, basic events, attributes, fault trees, event trees, end states, accident sequences, etc. For each relation, the following relation files exist: .BLK, .DAT, .DFL, and .IDX. Never delete these relation files; they are interrelated. Appendix A of the MAR-D Reference Manual[1] contains a more detailed description of these database relation files.

In addition to the relation files, SAPHIRE can also produce external *flat* or ASCII files. These can be extracted or loaded to or from the DOS family subdirectory using SAPHIRE software. These flat files, grouped according to the type of data they contain, are listed in Table 2-1. Once the data contained in the flat files have been entered into the SAPHIRE database, they are stored permanently in the relation files. Therefore, flat files can be deleted from the subdirectory to conserve disk space and later extracted from the interactive database if necessary. These flat files are useful to verify data entry for use in reports. Another important use is using an extracted file as a template to add additional data to the database.

There are two methods to create flat files. The first is to enter data into the interactive database using the IRRAS menus. Once the data are entered, the flat files containing this information can be extracted, as described in Appendix A. The second is to create and enter data into an ASCII flat file with the correct format and file name (as shown in Appendix B). These files can then be loaded into the interactive database, as described in Appendix A.

SAPHIRE also produces external report files. Report options are available in many sections of the software. The software allows the options to send reports to a printer, the computer screen, or to a file on any directory.

Note: Empty flat files can be extracted and serve as a template for the proper data entry format. These templates are available for those files listed in Table 2-1.

Table 2-1. SAPHIRE database file names and descriptions.

File name	Description	Applicable section in this manual
FAMILYNAME.FAD[a]	Family name and descriptions	4.2.3
FAMILYNAME.FAA	Family attributes	4.2.3
FAMILYNAME.FAT	Family text	4.2.3
FAMILYNAME.FAY	Family event tree recovery rules	—
FAMILYNAME.FAS	Family fault tree recovery rules	—
FAMILYNAME.FAP	Family endstate partition rules	—
FAMILYNAME.BED	Basic event names and descriptions	4.6.2
FAMILYNAME.BEI[a]	Basic event failure rates	4.6.3
FAMILYNAME.BEA	Basic event attributes	4.6.3
FAMILYNAME.FMD	Failure mode descriptions	Appendix B
FAMILYNAME.CTD	Component type descriptions	Appendix B
FAMILYNAME.STD	System type descriptions	Appendix B
FAMILYNAME.LCD	Location descriptions	Appendix B
FAMILYNAME.TTD	Train descriptions	Appendix B
FAULTTREE.DLS[a]	Fault tree graphics	4.5.1
FAMILYNAME.FTD	Fault tree names and descriptions	4.5.2
FAULTTREE.FTT	Fault tree text	4.5.2
FAMILYNAME.FTA	Fault tree attributes	Appendix B
FAMILYNAME.FTL[a]	Fault tree logic	4.5.1
FAMILYNAME.FTC[a]	Fault tree cut sets	4.5.4
FAULTTREE.FTY	Fault tree recovery rules	—
FAULTTREE.PID	Fault tree P&ID	—
EVENTTREE.ETG[a]	Event tree graphics	4.3.1
FAMILYNAME.ETD	Event tree names and descriptions	4.3.5
FAMILYNAME.ETT	Event tree text	4.3.5
FAMILYNAME.ETA	Event tree attributes	Appendix B
EVENTTREE.ETL[a]	Event tree logic	4.3.1
FAMILYNAME.ETR[a]	Event tree rules	4.3.4
EVENTTREE.ETY	Event tree recovery rules	—
EVENTTREE.ETP	Event tree end state partition rules	—
FAMILYNAME.ESD	End state names and descriptions	4.4.3
FAMILYNAME.EST	End state text	4.4.3
FAMILYNAME.ESC	End state cutsets	—
FAMILYNAME.SQD	Sequence names and descriptions	4.7.2
FAMILYNAME.SQC[a]	Sequence cut sets	4.7.1
FAMILYNAME.SQA	Sequence attributes	Appendix B
FAMILYNAME.SQT	Sequence text	4.7.2
FAMILYNAME.SQL	Sequence logic	4.3.6
FAMILYNAME.SQY	Sequence recovery rules	—
FAMILYNAME.SQP	Sequence end state partition rules	—
FAMILYNAME.GTD	Gate description	4.5.3
FAMILYNAME.GTA	Gate attributes	4.5.3
FAMILYNAME.CSD	Change set description	4.10.1/Appendix A
FAMILYNAME.CSI	Change set information	4.10.1/Appendix A

a. These files are required for obtaining sequence damage frequency. In other words, if only sequence frequency is of concern, these files must be loaded. However, to provide a complete data base for a specific family, all file listed must exist in the data base.

3. THE SAMPLE DATABASE

This section presents the sample database used to describe the data loading process in Section 4. Section 3.1 presents the basic assumptions concerning use of this manual. Sections 3.2 through 3.9 contain the actual data and a discussion of the sample database.

3.1 The Sample Database

Several assumptions concern the presentation of the sample database:

1. The SAPHIRE software has been loaded as described in Appendix A of the IRRAS Reference Manual.[2]

2. The user has a basic knowledge of using SAPHIRE to analyze event trees and fault trees.

3. The user has read the sections of the IRRAS Reference Manual[2] that provide an overview of the use of the software and the program menus, modules, and database concepts.

In the IRRAS Tutorial Manual,[3] a simple example shows the quantification for the frequency of getting to work. The tutorial leads the student through (a) the basic construction of event tree and fault trees, (b) entering basic event data, and (c) generation and quantification of both fault tree and sequence cut sets. In this report, a modification of the simple "getting to work" example is used to demonstrate the data loading process. Sections 3.2 through 3.9 present the sample database in a fashion similar to that found in a typical PRA. However, unlike most PRAs, the sample database contains only those data essential to constructing a workable database in SAPHIRE.

3.2 The Sample Database Event Tree

Using a fail-success logic, an event tree is developed to calculate the frequency that a worker will arrive on time, be late, or miss a day of work. The event tree (WORK) is shown in Figure 3-1. It was determined that the average working person is required to work approximately 248 days a year. In the WORK event tree, going to work was used as the initiating event (WORK). Initiating events are occurrences in a certain length of time that initiate a sequence of events. In this case, being required to get to work initiates the sequence of events leading to either getting to work on time, being late to work, or missing work completely.

The first event that should occur on a normal work day is that the alarm clock rings. Therefore, the first question to ask is "did the ALARM go off?" If it did not go off, then the worker will be late to work. If the alarm successfully wakes the worker, then some personal reason (i.e., sickness) may cause the worker to miss work. Therefore, the second question to ask is "did a PERSONAL reason make the worker miss work?" Thus, the ALARM may be successful but some PERSONAL reason may cause the worker to miss work. Now, either the alarm succeeded in waking the worker or the alarm failed and the worker woke up late, and if no personal circumstances cause the worker to miss work, then transportation problems may occur that cause the worker to be even later to work. Therefore, the third question to ask is "did the available transportation (TRNSPRT) fail?" Finally, if the alarm succeeded and no personal reasons interfered and transportation was available, then the worker will be successful in getting to work on time.

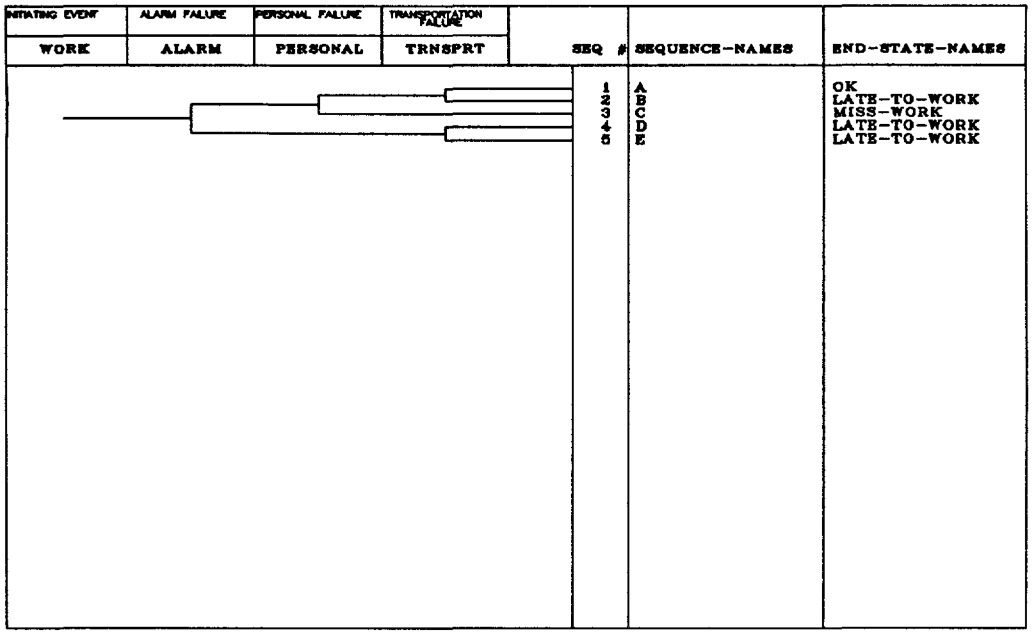


Figure 3-1. Going-to-work (WORK) event tree.

Assume that the probability of public transportation (represented by the top event TRNSPRT) will change depending on the time that the person attempts to use this service. This assumption implies that the probability of failing TRNSPRT is conditional on the time that the service is needed. Therefore, if ALARM fails then it is necessary to substitute a different fault tree or probability for the original TRNSPRT top event. The database has another new fault tree called TRNSPRT-2. This fault tree will contain a different probability for the basic event that represents the failure of the public transportation system when the demand for this service is at a later time.

The first four names along the topmost horizontal line of this figure represent the initiating event (WORK) and the top events (ALARM, PERSONAL, and TRNSPRT). Using the event tree in an analysis will enable the top events to be linked together. Standard practice depicts the initiating event as a horizontal line with systems connected in a branching structure, where an up branch indicates success and the down branch indicates failure. As the event tree logic is developed, a top event can either be passed (system not questioned) or questioned (system either succeeds or fails). Therefore, each pathway through an event tree has a combination of success, failure, or pass logic. This pathway is called a sequence. For example, following through the WORK event tree, sequence three (SEQ 3) is described as the success of ALARM, the failure of PERSONAL, and the pass of TRNSPRT.

3.3 The Sample Database Systems

Each of the top events presented in the WORK event tree may be further developed as system or fault tree logic. Fault tree analysis is a technique where many events (basic events) that interact to produce a complex event (top event) can be related using simple logical relationships (AND, OR, etc.). This permits the methodical building of a structure that can be used to analyze possible failures and to calculate the probability of failure. For this example, very simple fault trees (shown in Figures 3-2 through 3-5) were developed. These fault trees are used to determine the probability of each top event occurring and to develop fault tree and sequence cut sets.

The ALARM fault tree (see Figure 3-2) is a simple representation of modeling alarm clock failure. Some common reasons for alarm clock failure include setting the wrong time, failing to set the alarm, mechanical failure, or power failure (either battery or commercial). The OR-gate ALARM has three inputs, one OR-gate, one AND-gate, and one undeveloped event. The OR-gate ALARM-1 has two basic events as input representing the probability of setting the wrong time or failing to set the alarm. Either of these events, the alarm being set to the wrong time [ALM-SWT (alarm-set wrong time)] or the alarm not being set [ALM-FTS (alarm fail to set)], can fail the alarm clock. The undeveloped event under the OR-gate ALARM, ALM-MECH (ALARM-mechanical failure), will represent the probability of any of the mechanical functions associated with the alarm failing. Any mechanical failure will prevent the alarm from performing its function. The AND-gate ALARM-2 has two basic events as inputs representing the probability that power has failed to the alarm. It is necessary that both the commercial power [ALM-CPF (alarm-commercial power failure)] and the battery [ALM-BPF (alarm-battery power failure)] not work to fail the alarm.

The PERSONAL fault tree (Figure 3-3) is a simple representation modeling personal or human failure that results in missing work. Two of the most common reasons for failure include sickness or sickness in family. There are also many additional reasons for personal failure that are less common occurrences than sickness related failures. The OR-gate PERSONAL has three inputs; two basic events and one undeveloped event. The two basic events will represent the probability of either missing work due to being sick (SICK) or family illness [SICK-FAM (sick in family)]. The undeveloped event (OTHER) represents the probability that other personal reasons are responsible for failure.

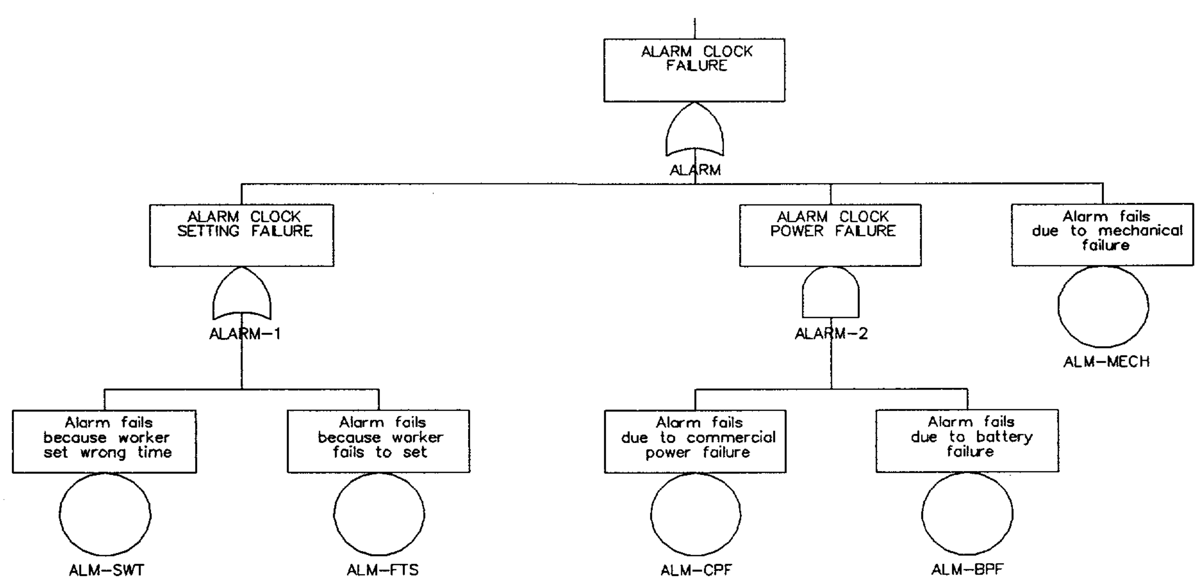


Figure 3-2. Alarm clock failure fault tree.

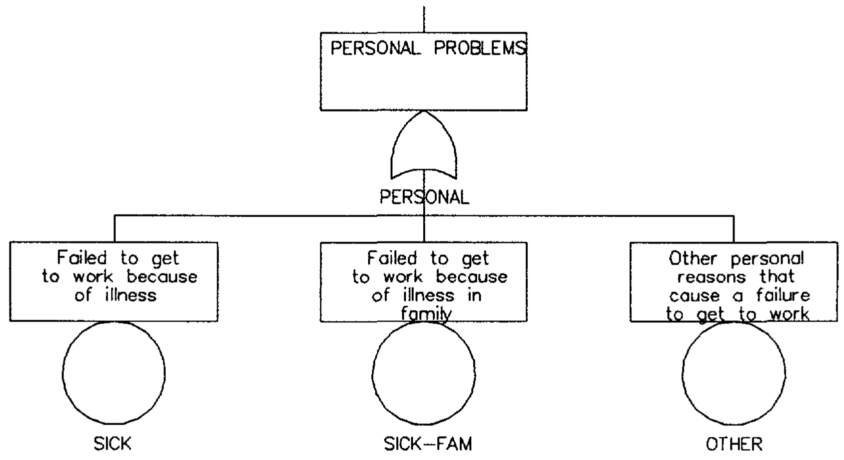


Figure 3-3. Personal problems fault tree.

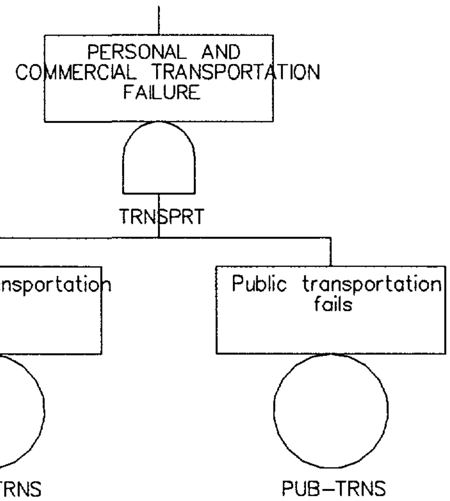


Figure 3-4. Transportation failure fault tree (normal time frame).

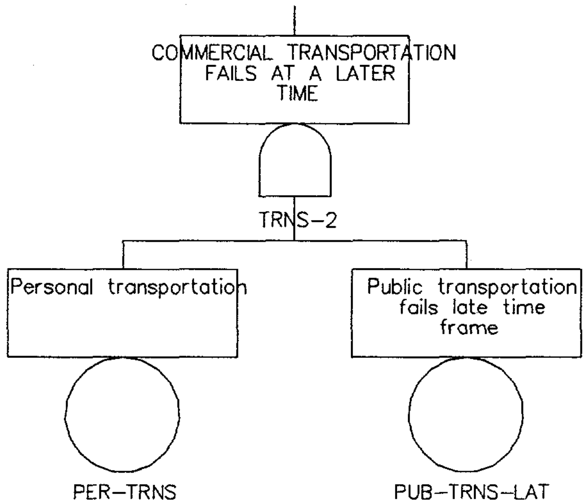


Figure 3-5. Transportation failure fault tree (late time frame).

The third fault tree TRNSPRT (Figure 3-4) is a simple representation modeling transportation failure. Two common modes of transportation include personal (such as a car) and public (such as a bus or train). The AND-gate TRNSPRT has two basic events as inputs. The two basic events will represent the probability of public transportation [PUB-TRNS (public transportation)] and personal transportation [ER-TRNS (personal transportation)] failure.

An additional fault tree TRNSPT-2 (Figure 3-5) is a modification of the TRNSPRT fault tree. Since the probability of obtaining public transportation is dependent upon the time of day, this fault tree is a representation modeling transportation at a later time. In this situation, the probability of public transportation failing is less due to the later time frame. Then, if ALARM fails, the worker needs public transportation later than if the ALARM had succeeded. In this scenario, it is necessary to substitute a fault tree for the TRNSPRT top event (TRANSPT-2) that contains the probability of failure of the public transportation system in a later time frame.

3.4 The Sample Database Basic Events

Information on the basic event values and descriptions for the sample problem is provided in Tables 3-1 and 3-2. Table 3-1 provides the necessary basic event and initiating event information to duplicate the analysis performed on this problem. Typically, PRAs contain more basic event information (e.g., system type, failure mode) than will need to be entered into the database to complete the analysis. Note that the uncertainty value contained in Table 3-1 is the lognormal distribution error factor.

Table 3-1. Basic event values for the sample problem.

Basic event	Distribution type	Calculation type	Mean value	Uncertainty value
ALM-BPF	Lognormal	1	9.0E−4	3
ALM-CPF	Lognormal	1	1.5E−2	3
ALM-FTS	Lognormal	1	5.5E−3	10
ALM-MECH	Lognormal	1	2.7E−4	3
ALM-SWT	Lognormal	1	2.7E−3	10
MEDICINE	Lognormal	1	8.1E−3	5
OTHER	Lognormal	1	5.0E−1	10
TRNS	Lognormal	1	2.7E−3	3
PUB-TRNS-2	Lognormal	1	2.0E−3	3
SICK	Lognormal	1	8.1E−3	10
SICK-FAM	Lognormal	1	4.0E−3	10
WORK	Lognormal	1	2.48E+2/yr	10

Table 3-2. Basic event descriptions for the sample problem.

Basic event	Description
ALM-BPF	Alarm fails due to battery failure
ALM-CPF	Alarm fails due to commercial power failure
ALM-FTS	Alarm fails because worker failed to set alarm
ALM-MECH	Alarm fails due to mechanical failure
ALM-SWT	Alarm fails because worker set wrong time
MEDICINE	Recovery for sickness preventing attending work
OTHER	Other personal reasons that cause a failure to get to work
PER-TRNS	Personal transportation
PUB-TRNS	Public transportation fails
PUB-TRNS-2	Public transportation fails late time frame
SICK	Failed to get to work because of illness
SICK-FAM	Failed to get to work because of illness in family
WORK	Event tree (WORK) initiating event

Since the sample database is simple compared to traditional PRA databases, no advanced external event analysis features are covered. Consequently, fire, flood, and seismic analysis are not directly addressed by way of the sample database. However, details for data loading and manipulation for seismic analysis are presented in Appendix D.

3.5 Sample Database System Cut Sets

The following are the system cut sets and minimum cut set (mincut) upper bound for those systems contained in the sample database. The system modeling of "personal failure due to sickness and other reasons" has the greatest probability of occurring.

System: **ALARM**
Mincut Upper Bound: 2.705E-3

Cut No.	Total (%)	Set (%)	Probability	Cutsets
1	99.8	99.8	2.7E-3	ALM-SWT
2	100.0	0.2	5.5E-6	ALM-FTS
3	100.0	0.0	2.7E-8	ALM-MECH
4	100.0	0.0	1.3E-9	ALM-BPF, ALM-CPF

System: **PERSONAL**
Mincut Upper Bound: 2.007E-2

Cut No.	Total (%)	Set (%)	Probability	Cutsets
1	40.3	40.3	8.1E-3	OTHER
2	80.7	40.3	8.1E-3	SICK
3	100.0	19.9	4.0E-3	SICK-FAM

System: **TRNS-2**
Mincut Upper Bound: 1.100E-5

Cut No.	Total (%)	Set (%)	Probability	Cutsets
1	100.0	100.0	1.1E-5	PER-TRNS, PUB-TRNS-LAT

System: **TRNSPRT**
Mincut Upper Bound: 1.485E-5

Cut No.	Total (%)	Set (%)	Probability	Cutsets
1	100.0	100.0	1.4E-5	PER-TRNS, PUB-TRNS

3.6 Sample Database Sequence Cut Sets

The following are the cut sets and frequencies for the sequences from the WORK event tree. Since Sequence 1 represents successfully getting to work, it is not presented. Sequence 3 is the largest and only contributor to missing work. Sequence 4 is the largest contributor to being late-to-work.

Sequence: 2 (calculated frequency = 3.682E-3/yr)

Cutset	Frequency	Cutset
1	3.6E-3	WORK, PER-TRNS, PUB-TRNS

Sequence: 3 (calculated frequency = 4.0/yr)

Cutset	Frequency	Cutset
1	2.0E+0	WORK, OTHER
2	1.0E+0	WORK, SICK, MEDICINE
3	9.9E-1	WORK, SICK-FAM

Sequence: 4 (calculated frequency = 6.709E-1/yr)

Cutset	Frequency	Cutset
1	6.7E-1	WORK, ALM-SWT
2	1.3E-3	WORK, ALM-FTS
3	6.7E-6	WORK, ALM-MECH
4	3.3E-7	WORK, ALM-BPF, ALM-CPF

Sequence: 5 (calculated frequency = 7.380E-6/yr)

Cutset	Frequency	Cutset
1	7.3E-6	WORK, ALM-SWT, PER-TRNS, PUB-TRNS-LAT
2	1.5E-8	WORK, ALM-FTS, PER-TRNS, PUB-TRNS-LAT
3	7.3E-11	WORK, ALM-MECH, PER-TRNS, PUB-TRNS-LAT
4	3.6E-12	WORK, ALM-BPF, ALM-CPF, PER-TRNS, PUB-TRNS-LAT

3.7 Sample Database Recovery Actions

Sequence 3 shown in the sequence cut set list in Section 3.6 accounts for most of the days lost at work (4.0 times per year). Notice that a basic event, MEDICINE, has been added to the cutset containing sick. It was anticipated that 50% of the time it may be possible that an individual will take medicine and feel well enough to attend work. MEDICINE is the recovery action added *after* the sequence cutset generation.

3.8 Sample Database Uncertainty

The following summarizes the sequence, system, and endstate uncertainty. All uncertainties were performed using a Monte Carlo simulation with 1000 samples. Table 3-3 lists the uncertainty results for the systems. Table 3-4 lists the uncertainty results for the sequences. Table 3-5 lists the uncertainty results for the end state analysis.

Table 3-3. System uncertainty values report.

No.	System name	Mean Median	MinCut upper bound Standard deviation	5th Percentile 95th Percentile	Minimum Maximum
1	ALARM	2.932E-003 1.157E-003	2.705E-003 7.036E-003	9.971E-005 1.033E-002	1.714E-005 1.205E-001
2	PERSONAL	2.007E-002 1.216E-002	2.007E-002 2.630E-002	2.767E-003 6.382E-002	5.473E-004 3.664E-001
3	TRNS-2	1.504E-005 7.385E-006	1.485E-005 2.968E-005	1.141E-006 5.289E-005	3.449E-008 4.646E-004
4	TRNSPRT	2.932E-003 1.157E-003	2.705E-003 7.036E-003	9.971E-005 1.033E-002	1.714E-005 1.205E-001

Table 3-4. Sequence uncertainty values report.

No.	Event tree name	Mean Median	MinCut upper bound Standard deviation	5th Percentile 95th Percentile	Minimum Maximum
1	WORK	3.699E-003 1.658E-003	3.682E-003 7.072E-003	2.336E-004 1.262E-002	5.091E-005 9.751E-002
2	WORK	4.303E+000 1.959E+000	3.985E+000 9.597E+000	3.729E-001 1.279E+001	8.912E-002 1.813E+000
3	WORK	7.012E-001 2.348E-001	6.709E-001 1.578E+000	2.007E-002 2.677E+000	2.867E-003 1.839E+001
4	WORK	1.171E-005 1.232E-006	7.380E-006 1.307E-004	4.896E-008 3.014E-005	1.073E-009 3.977E-003

Table 3-5. End state uncertainty values report.

Endstate name	Mean Median	MinCut upper bound Standard deviation	5th Percentile 95th Percentile	Minimum Maximum
LATE-TO-WORK	7.343E-001 2.480E-001	6.746E-001 2.442E+000	2.704E-002 2.370E+000	2.408E-003 4.257E+001
MISS-WORK	4.269E+000 2.324E+000	3.985E+000 6.613E+000	3.985E-001 1.449E+001	5.973E-002 6.807E+001

3.9 Sample Database Importance

The following is a report on the Fussell-Vesely Importance Measure for each basic event over the total database analysis. Table 3-6 shows the results of the importance analysis for the sample database.

Table 3-6. Results of sample database importance analysis.

Event name	Number of occurrences	Probability of failure	Fussell-Vesely importance
WORK	12	2.480E+002	1.000E+000
OTHER	1	8.100E-003	4.276E-001
MEDICINE	1	5.000E-001	2.129E-001
SICK	1	8.100E-003	2.129E-001
SICK-FAM	1	4.000E-003	2.103E-001
ALM-SWT	2	2.700E-003	1.437E-001
PER-TRNS	5	5.500E-003	7.919E-004
PUB-TRNS	1	2.700E-003	7.904E-004
ALM-FTS	2	5.500E-006	2.919E-004
PUB-TRNS-LAT	4	2.000E-003	1.584E-006
ALM-MECH	2	2.700E-008	1.433E-006
ALM-BPF	2	9.000E-008	7.166E-008
ALM-CPF	2	1.500E-002	7.166E-008

4. LOADING THE SAMPLE DATABASE

4.1 Introduction

This section describes the process of loading the sample database presented in Section 3. The section is organized to reflect the methodology that has proven useful while working with actual PRA data. The section organization is as follows:

Section 4.2	Adding and Selecting the Database Family
Section 4.3	Loading the Event Tree Data
Section 4.4	Loading Endstate Data
Section 4.5	Loading the Fault Tree Data
Section 4.6	Loading Basic Event Data
Section 4.7	Loading Sequence Data
Section 4.8	Recovery Actions
Section 4.9	Generating Uncertainty
Section 4.10	Additional Features.

Each section presents methods used for entering a specific type of data (there may be several methods possible). The merits of each data entry method is discussed and a brief sketch of the actual steps used to enter the data using this method is presented. Manuals and guides that may add useful information to the method are also cited.

4.2 Adding and Selecting the Database Family

The necessary first step in loading a database is adding and selecting the family that will contain the database. Adding and selecting the database family includes

1. Adding the family (Section 4.2.1)

2. Selecting the family (Section 4.2.2)

3. Entering family information and text (Section 4.2.3)

4. Extracting and verifying the family flat files (Section 4.2.4).

4.2.1 Adding the Family

The SAPHIRE database structure is divided into families Since access to the SAPHIRE interactive database is obtained through the appropriate family, a family is the first thing that must be added. A family is any logical grouping of fault trees and event trees with their associated basic events, cut sets, reliability data, and descriptions. The family concept allows separation of any number of distinct databases. When a database family is created in one of the SAPHIRE programs, a corresponding DOS subdirectory contained in the SAF50 subdirectory is also created.

There is only one method to add a family and this can be accomplished in either SARA, IRRAS, or MAR-D. The procedure is shown in detail in Appendix A. The procedure requires:

1. Entering the Modify Data Base module from the main menu
2. Selecting the Family option
3. Selecting the Add option
4. Typing the family name (SAMPLE) and other family data into the Add Family form shown in Figure 4-1
5. Enter "saving the family."

```
                     A d d   F a m i l y

            Option |A|  Exit / Add / Passwords

     Name                        Type                 Tree Type
     Location                    Design
     Company                     Vendor
     Description

        Seismic Histograms
     Low                         Operational date   ----/--/--
     Medium                    Qualification date ----/--/--
     High                          Mission time       ----------

             Press <F1> for Help or List of Values
```

Figure 4-1. The IRRAS Add Family menu screen.

4.2.2 Selecting the Family

Once the family has been added, the next step is to *SELECT* this database for data entry and analysis. It is possible to modify or enter any information concerning the family (e.g., family text) from the Family option under the Modify Data Base module. It is not possible to extract files concerning this family or enter any other data information (e.g., basic events) until this family has been selected. The database must be selected from within the SAPHIRE programs to allow entry of data other than family data. Since SAPHIRE is an interactive program, once the family is selected it will remain selected for all the programs (e.g., IRRAS, SARA) even if a program is exited. Note that the name of the family selected generally appears in the upper left hand corner of most SAPHIRE menus.

There is only one method to select a family and this can be accomplished in either SARA, IRRAS, or MAR-D. The procedure is shown in detail in Appendix A.

The procedure requires (1) entering the Select Family module from the main menu and (2) highlighting the family to select (SAMPLE) and then pressing <enter>.

4.2.3 Entering Family Information, Description, and Text

Family description, information, and text can also be entered into the database whether the family has been selected or not. To add the family description ("This is sample database") or the family text ("A simple example that models the probability of getting to work on time"), enter the Family option under the Modify Data Base module from the main menu. It is possible to enter this family information at the time that the family is added using the menu shown in Figure 4-1. This family information includes location, company, type, design, vendor, tree type, seismic histograms, operational date, qualification data, and mission time. The family information (e.g., description and text) can also be entered after the family has been added by selecting Modify in the Family option under the Modify Data Base module from the IRRAS main menu. This form is very similar to the Add Family menu shown in Figure 4-1.

Methods for Entering Family Information, Description, and Text.

1. The first step is to use the interactive database to enter the data. The procedure for using the interactive database is as follows:

    a. Select the Modify Data Base option from the main IRRAS menu.

    b. Select the Family option.

    c. From this menu, either select the Modify option to enter either family information and descriptions or the Text option to enter family text. Both help screens and other information in these menus allow the user to follow the procedure to enter data.

2. Second, the .FAA, .FAD and .FAT family flat files can be extracted as shown in Table 4-1 and used as a template to enter information using a text editor. The procedure for using extracted flat files is as follows:

   a. Extract the family files, .FAA, FAD, and .FAT (the extract and load processes are described in detail in Appendix A).

   b. Use an editor to modify and add the family data to the extracted files. A detailed description of the flat file format is available in Appendix B.

   c. Load the finalized files back into the interactive database. The interactive database should now contain the family data.

4.2.4 Extracting and Verifying the Family Data

It is often necessary to verify that data are accurate. The SAPHIRE flat files are particularly useful for this task. The flat files extracted from the sample database (shown in Table 4-2) can be used to verify the family information entered in the interactive database. Notice that not all the possible entry fields (e.g., Design) have been filled. Many options are provided in SAPHIRE that may not be applicable to every database, and, subsequently, some areas may be blank.

4.3 Loading the Event Tree Data

The next step in loading a database is to enter the database event trees and verify their accuracy. The event tree data entry is complicated by the fact that the IRRAS software uses an interactive database. Information entered during the process of graphical event tree construction will appear in other areas of the program. Those event trees that contain an initiating event will be listed in the Link Event Tree option (in the Create event trees and Analyze sequences modules) and the Event Trees option (in the Modify Data Base module). Event trees with or without an initiating event will appear in the Plot, Graphic load, and Extract event trees options (in the Create Event Trees module). Top events are listed as both systems and developed basic events in the Modify Data Base module, while initiating events are only listed as basic events. The information in any of these internal lists can subsequently be extracted into IRRAS flat files.

Table 4-1. Extracted family flat files.

File	Extracted file information
.FAA	SAMPLE = * Name , Mission, NewSum, Company, Location, Typ, Design, Vendr, Arch Eng, OpDate, QualDate SAMPLE, 2.400E+001,-----E---, , , , , , , ,----/--/--
.FTD	SAMPLE ,
.FAT	SAMPLE =

Table 4-2. Extracted finalized family flat files.

File	Extracted file information
.FAA	SAMPLE = * Name , Mission, NewSum, Company, Location, Typ, Design, Vendr, Arch Eng, OpDate, QualDate SAMPLE , 2.40E+001, ----E--, STANDARD, HOMETOWN , , , , ,----/--/--,----/--/--
.FTD	SAMPLE ,This is sample data base
.FAT	SAMPLE = A simple example that models the probability of getting to work on time.

It is not necessary to enter the event trees at this point, but it has proven to be the most efficient method for entering nuclear power plant PRAs. The sample database event tree to be loaded is shown in Figure 4-2.

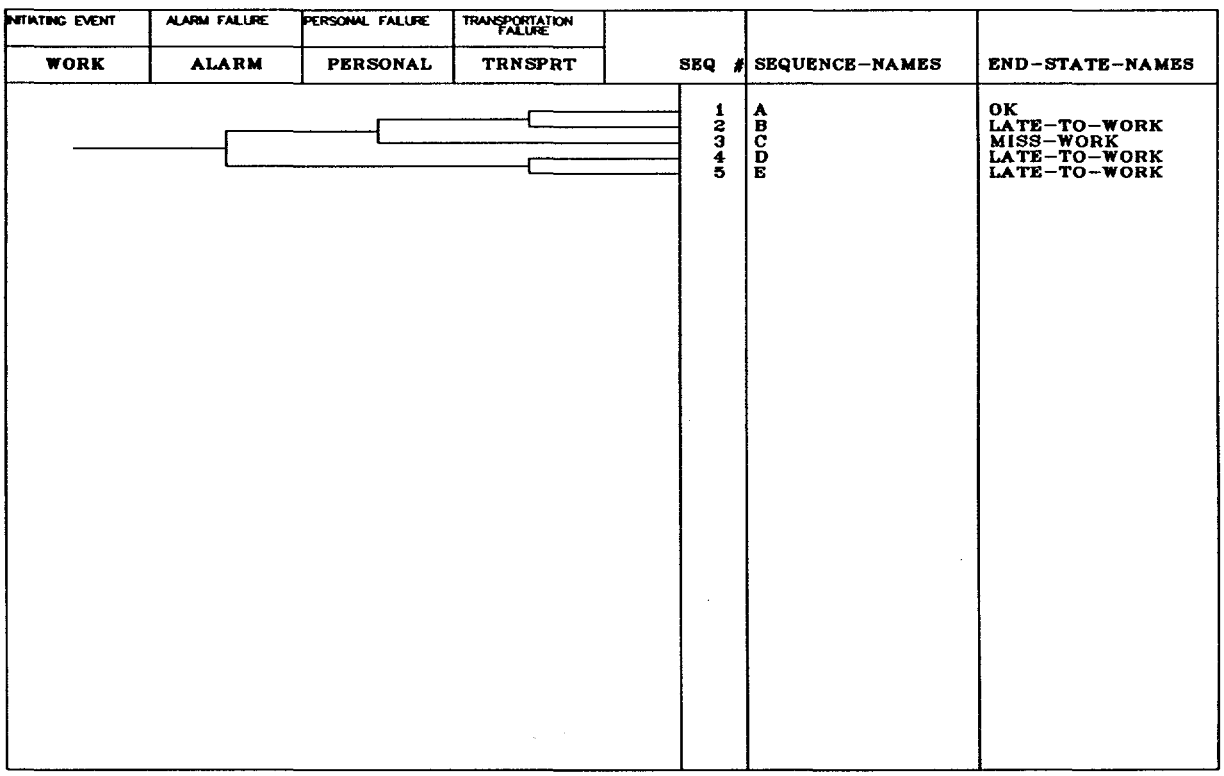


Figure 4-2. Going to work event tree graphic.

The process of loading an event tree includes

1. Entering the event tree logic (Section 4.3.1)

2. Entering sequence names in graphics (Section 4.3.2)

3. Entering top event descriptions (Section 4.3.3)

4. Entering substitutions (if-then) rules (Section 4.3.4)

5. Entering event tree descriptions and text (Section 4.3.5)

6. Generating and verifying the event tree logic (Section 4.3.6).

4.3.1 Entering the Event Tree Logic

In the sample database, event tree logic is used as the basis for linking system fault trees and generating sequence logic to ultimately generate sequence cut sets. Some types of databases may not use event trees, but they are typically used to varying degrees in PRA methodology. SAPHIRE was originally designed to handle the more common type of approach, the large fault tree/small event tree approach, represented by the sample database. Other databases may use the large event tree/small fault tree approach. Version 5.0 of the SAPHIRE suite has been adapted to better handle the large event tree type of database. Note that SAPHIRE does not accept other software event tree graphics; therefore, each event tree will have to be created individually.

Methods for Entering the Event Tree Logic.

1. The most efficient method to load event tree logic is to enter the event tree structure information into SAPHIRE in the graphics module. It is straightforward to enter event tree logic into the graphics module, and the process of entering and saving an event tree similar to the sample database is discussed in great detail in the IRRAS Tutorial Manual.[3]

   The event tree creation procedure requires the user to (a) enter the Create Event Trees module from the IRRAS main menu, (b) enter the Create Event Trees Option, and (c) enter the basic event tree structure, as shown in Figure 4-2, using the graphics editor.

2. It is possible, but may be difficult, to enter the event tree graphic logic into a flat file and then load this file into SAPHIRE. As the development of the small WORK event tree is presented in the following sections, it will be obvious that the more highly branched the event tree becomes, the more confusing the resulting logic. Therefore, this method is not discussed.

Once an event tree is created, any of the flat files for this tree can be extracted. At this stage, many of the extracted flat files shown in Table 2-1 (Section 2.4) will not contain data other than the event tree and family name. The flat files that will contain data are the event tree graphics (.ETG) and the event tree logic (.ETL) shown in Table 4-3. These two files are identical in SAPHIRE.

Table 4-3. Extracted event tree flat files (with logic only).

Files	Extracted file information
.ETG	SAMPLE, WORK, WORK =
	^TOPS
and	ALARM, PERSONAL, TRNSPRT
	^LOGIC
.ETL	+1 +2 +3
	-3
	-2 3
	-1 2 +3
	-3
	^SEQUENCES
	N, SEQUENCE-NAMES, N, END-STATE-NAMES, N, FREQUENCY, N, EXTRA,
	Y, A, Y, , Y, , Y, ,
	Y, B, Y, , Y, , Y, ,
	Y, C, Y, , Y, , Y, ,
	Y, D, Y, , Y, , Y, ,
	Y, E, Y, , Y, , Y, ,
	^PARMS
	START 80.96 WINDOW 0.00, 0.00, 120.00, 120.00
	HEADER 86.40, 111.60, 136.80, 162.00
	STRING E
	DEFFONT 5
	TOPWIDTH 8
	TOPSIZE 1.44
	TOPFONT 1
	DESHITE 0
	DESSIZE 0.50
	DESFONT 5
	NODEHITE 2.00
	ENDSIZE 1.50
	ENDFONT 1

Note:

- Although 16 characters are allowed for a top event name, if an associated fault tree will be linked to this top event, limit the top event name to eight characters.
- When saving an event tree graphics file, verify that the file name under EVENT matches the file name under SAVE. The EVENT and SAVE options are in the FILE option of Create Event Trees.
- Remember that event trees cannot transfer to the middle of other event trees.
- Whenever possible, for ease of identification, identify initiating events by prefixing their names with the letters IE; for example, IE-xx.
- Give all event trees unique names for identification and tracking. It may be useful to include the family name, the event tree name, and the document-related page number.

4.3.2 Entering Sequence Names in Graphics

Event tree sequence names are automatically numbered, and a default name is entered when an event tree is created. Being able to assign sequence names allows the user the option of using a specific naming scheme. Sequence data will appear in the event tree graphics flat files as shown in Table 4-4. See Table 2-1, Section 2.4, for a list of all flat files that can be extracted. Note that even though the sequence names may appear in the graphics, they will not appear as extractable flat files until the sequences have been generated and quantified.

Methods for Entering Sequence Names in Graphics.

1. Sequence names can be entered in the graphics editor. This is potentially the most time-consuming method but is the most straightforward. In this case, the event tree could be finalized and files extracted without any intermediate steps. Procedures for adding end state and sequence names using the graphics editor are provided in the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual.[3]

   The procedure for entering the sequence names in the graphics menu requires the user to (a) enter the Create Event Trees option from the main menu, (b) access the Create Event Tree graphics, (c) use the Edit Header option in the ENDS popup menu to turn on the sequences, and (d) use the End States option in the ENDS popup menu to enter data.

2. Sequence names can also be entered using the sequence editor. This is perhaps the easiest method as it does not require leaving the IRRAS environment. Though it may be slower than the third method discussed here, it is recommended for most situations. One potential problem is that the headers cannot be toggled on and off in the sequence editor; therefore, even though end state or sequence names have been added, they will not appear in the graphics display. Indepth procedures for adding end state and sequence information using the sequence editor is provided in the IRRAS Tutorial Manual.

Table 4-4. Extracted event tree flat files (with logic and sequence names).

File	Extracted file information
.ETG	SAMPLE, WORK, WORK =
	^TOPS
or	ALARM, PERSONAL, TRNSPRT
	^LOGIC
.ETL	+1 +2 +3
	-3
	-2 3
	-1 2 +3
	-3
	^SEQUENCES
	N, SEQUENCE-NAMES, Y, END-STATE-NAMES, N, FREQUENCY, N,EXTRA,
	Y, A, Y, OK, Y, , Y, ,
	Y, B, Y, LATE-TO-WORK, Y, , Y, ,
	Y, C, Y, MISS-WORK, Y, , Y, ,
	Y, D, Y, LATE-TO-WORK, Y, , Y, ,
	Y, E, Y, LATE-TO-WORK, Y, , Y, ,
	^PARMS
	START 80.96 WINDOW 0.00, 0.00, 120.00, 120.00
	HEADER 86.40, 111.60, 136.80, 162.00
	STRING E
	DEFFONT 5
	TOPWIDTH 8
	TOPSIZE 1.44
	TOPFONT 1
	DESHITE 0
	DESSIZE 0.50
	DESFONT 5
	NODEHITE 2.00
	ENDSIZE 1.50
	ENDFONT 1

The procedure for entering the sequence names using the sequence editor requires the following:

a. Entering the Analyze Sequence option from the main menu.

b. Selecting the Link Event Trees option.

c. Highlighting the WORK event tree.

d. Selecting the Sequence Editor.

e. Use the Line Edit option to enter the sequence name data. Additional discussion of this feature is presented in the IRRAS Reference Manual.[2]

3. The sequence names can also be entered into the event tree flat file (.ETG or .ETL) that was extracted from the IRRAS program using a text editor. After modification, both files must be loaded, as described in Appendix A. This may be the fastest method available but requires more substantial steps and is prone to errors. Therefore, this method is not discussed further.

Note:

- If large event trees are created, the user may not want to process large numbers of sequence cut sets. To avoid this, place an @ symbol as the first character in the endstate name column. This will cause IRRAS to ignore these sequences during the cut set generation process.

- Any sequence with an OK, Ignore, or Success entered as the endstate name will not be generated.

  Caution: If the @ symbol is used in front of all sequences, an error message will be displayed if an attempt is made to view the event tree rules.

4.3.3 Entering Top Event Descriptions

Descriptions of top events are commonly found, as was shown in Figure 4-2. The descriptions normally appear above the top event designator. Top event descriptions can either be added in the graphics option or the .ETG can be modified using a text editor. Table 4-5 shows the event tree graphics flat files containing the top event descriptions.

Table 4-5. Extracted event tree flat files (with logic, sequence names, and top event descriptions).

File	Extracted file information
.ETG	SAMPLE, WORK, WORK =
	^TOPS
and	ALARM, PERSONAL, TRNSPRT
	^LOGIC
.ETL	+1 +2 +3
	-3
	-2 3
	-1 2 +3
	-3
	^SEQUENCES
	N, SEQUENCE-NAMES, N, END-STATE-NAMES, N, FREQUENCY, N, EXTRA-#1,
	Y, A, Y, , Y, , Y, ,
	Y, B, Y, , Y, , Y, ,
	Y, C, Y, , Y, , Y, ,
	Y, D, Y, , Y, , Y, ,
	Y, E, Y, , Y, , Y, ,
	^TOPDESC
	"INITIATING EVENT"
	!
	"ALARM FAILURE"
	!
	"PERSONAL FAILURE"
	!
	"TRANSPORTATION"
	"FAILURE"
	!
	^PARMS
	START 80.96 WINDOW 0.00, 0.00, 120.00, 120.00
	HEADER 86.40, 111.60, 136.80, 162.00
	STRING E
	DEFFONT 5
	TOPWIDTH 8
	TOPSIZE 1.44
	TOPFONT 1
	DESHITE 3
	DESSIZE 0.50
	DESFONT 5
	NODEHITE 2.00
	ENDSIZE 1.50
	ENDFONT 1

Methods for Entering Top Event Descriptions.

1. Top event descriptions are easily entered in the graphics editor. This is potentially the most time consuming but the most straightforward. In this case, the event tree is finalized and files extracted without any intermediate step. Indepth procedures for adding top event descriptions using the graphics editor is provided in the IRRAS Tutorial Manual.[3]

   The procedure for entering the top event description into event tree graphics is to (a) enter the Create Event Trees option from the main menu, (b) access the Create Event Tree graphics, and (c) enter the top event descriptions by selecting TOPS, DESC, LINE, and EDIT. The IRRAS Reference Manual[2] and IRRAS Tutorial Manual[3] contain details concerning this process.

2. Top event descriptions also can be entered into the event tree flat file (.ETG or .ETL) that was extracted from the IRRAS program using a text editor (see Table 4-5). After modification, both files must be loaded as described in Appendix A. This may be the fastest method available but requires more substantial steps and is prone to errors since the information needs to be reloaded. This method is not discussed further.

4.3.4 Entering If-Then Rules

Substitutions of different fault trees or top event probabilities are very commonly used in event tree logic. In this sample problem, for example, there may be a different probability of failure for the transportation, depending on whether the alarm succeeds or fails. As discussed in Section 3.1, this is due to the increased availability of later public transportation. SAPHIRE uses if-then rules to allow substitutions of event tree top events. Table 4-6 shows the sample database .ETR file.

Methods for Entering If-then Rules.

1. If-then rules can be entered from the Rule Editor menu. This is the most straightforward and simplest, particularly when the rules are short.

   The procedure for entering the if-then rules is to (a) enter the Analyze Event tree module from the IRRAS main menu, (b) access the Link Event Tree option, and (c) enter rules as directed by the editor screen.

2. If-then rules can also be entered into an event tree rule flat file using the SAPHIRE format. After the file is developed, it is necessary to load this file. (The loading procedure is discussed in Appendix A.) This method may be the fastest (particularly with a large group of rules) but requires more substantial steps and is prone to errors since the rule information needs to be reloaded. This method is not discussed further. Note that once a rule has been entered for an event tree, the .ETR flat file can be extracted for use as a template for subsequent rules.

Table 4-6. Extracted event tree rules flat file.

File	Extracted file information
.ETR	SAMPLE, WORK= SET , Rule to substitute TRNSPRT-2 for TRNSPRT IF ALARM THEN TRNSPRT = TRNSPRT-2.

4.3.5 Entering Event Tree Descriptions and Text

Many PRAs contain descriptions and extensive text concerning the event trees in the analysis. The sample database has an event tree description (WORK EVENT TREE) and text for demonstration purposes. Table 4-7 shows the extracted flat files containing the descriptions and text.

Methods for Entering Event Tree Descriptions and Text.

1. Event tree descriptions and text can be entered in the Modify Data Base module. This is perhaps the easiest method since there are usually a limited number of event trees and it does not require leaving the IRRAS environment. Though it may be slower than the other method discussed here, it is recommended for most situations. Procedures for adding descriptions and text are in the IRRAS Reference Manual.[2]

   The procedure for entering the event tree descriptions and text is to (a) enter the Modify Data Base module from the main menu, (b) select the Event Tree option, and (c) select the Modify option to enter the event tree description or the Text option to enter event tree text.

2. The description data can be entered into the event tree description flat file (.ETD) extracted from the IRRAS program, using a text editor. The event tree textual data can be entered into the event tree text flat file (.ETT) using the SAPHIRE format. (This is also true of the .ETD). After modification or development, both files must be loaded as described in Appendix A. This method is not discussed further.

Table 4-7. Extracted event tree description and text flat files.

File	Extracted file information
.ETD	SAMPLE = WORK ,WORK EVENT TREE
.ETT	SAMPLE, WORK= A FAIL-SUCCESS LOGIC WAS USED TO DEVELOP AN EVENT TREE TO CALCULATE THE FREQUENCY THAT THE AVERAGE PERSON WILL ARRIVE ON TIME, BE LATE OR MISS A DAY OF WORK.

4.3.6 Generating and Verifying Event Tree Logic

Basic event tree logic is verified using either the graphics visual picture or by generating sequence logic and examining the results of the sequence generation process. It is recommended that both these processes be performed after creating an event tree and entering all the associated data. The sequence logic flat file is shown in Table 4-8. The methods discussed below allow verification of all the data entered, as described in Section 4.3.

Methods for Generating and Verifying Event Tree Logic.

1. A graphical output can be obtained for each event tree. This graphic output can be sent directly to a printer, or either a raster or hpgl file can be produced. Note that the graphical output can be verified as accurate, but any if-then rules will need to be examined. It is recommended that a printout be produced for each event tree to verify entry. Procedures for producing a printout are available in the IRRAS Reference Manual.[2]

   The procedure for obtaining a printout requires the user to (a) enter the Create Event Trees module from the main menu, (b) select the Plot Trees option, and (c) highlight the event tree and select the option of choice (HPGL, Raster, or Plotter files).

2. In the process of generating sequences, event tree logic can be verified. This process produces the sequence logic that will be used by the interactive database to produce sequence cutsets.

   The procedure for generating sequences and obtaining a printout for verification requires the following:

   a. Entering the Analyze Sequences module from the main menu.

   b. Selecting the Link Event Trees option.

   c. Highlighting the event tree and selecting Generate Sequences. When Generate Sequences is chosen, an intermediate screen will appear that queries whether the sequence generation information should be sent to a report on the screen, to the printer, or a file on the hard disk drive. The output will be similar to the extracted flat file shown in Table 4-8.

Table 4-8. Extracted sequence logic flat files.

File	Extracted file information
.SQL	SAMPLE, WORK, 2= /ALARM /PERSONAL TRNSPRT . ^EOS SAMPLE, WORK, 3= /ALARM PERSONAL . ^EOS SAMPLE, WORK, 4= ALARM /TRNS-2 . ^EOS SAMPLE, WORK, 5= ALARM TRNS-2 .

4.4 Loading Endstate Data

This section describes loading the endstate data so that endstate data are included in both the graphics and analysis portion of SAPHIRE. The following steps must be performed to actually load and verify the endstate data:

1. Entering endstate names in graphics (Section 4.4.1)

2. Entering endstates for analysis (Section 4.4.2)

3. Entering the endstate description and text (Section 4.4.3).

4.4.1 Entering Endstate Names in Graphics

Endstate data are used in a PRA analysis to group sequences that have similar outcomes for subsequent entry into the level 2 analysis. The sample database has four sequences that are grouped into two endstates (late-to-work and miss-work). A subsequent analysis is possible on these two endstates. There are two flat files that can be obtained that contain endstate data, as shown in Table 2-1, Section 2.4, and Table 4-9.

Table 4-9. Extracted endstate flat files.

File	Extracted file information
.ESD	SAMPLE = LATE-TO-WORK , This end state represents being late to work MISS-WORK , This end state represents missing work
.EST	SAMPLE, LATE-TO-WORK= THIS IS THE LATE TO WORK END STATE

Methods for Entering Endstate Names in Graphics.

1. Endstate names may be entered in the graphics editor. Using the graphics editor is potentially the most time-consuming but the most straightforward method. In this case, the event tree could be finalized and files extracted without any intermediate step. Indepth procedures for adding end state and sequence names using the graphics editor is provided in Appendix A. Both the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] contain details concerning this process.

   The procedure for entering the endstate name in the graphics editor requires the following:

   a. Entering the Create Event Trees option from the main menu

   b. Accessing the Create Event Tree graphics

   c. Using the Edit Header option in the ENDS popup menu to turn on the endstates by changing the Ns to Ys

   d. Using the End States option in the ENDS popup menu to enter data.

2. Endstate names also can be entered using the sequence editor. This is perhaps the easiest method as it does not require leaving the IRRAS environment. Though it may be slower than the third method discussed here, it is recommended for most situations. One potential problem is that the headers cannot be toggled on and off in the sequence editor, and, even though end state or sequence names have been added, they may not automatically appear in the graphics display. Indepth procedures for adding end state and sequence information using the sequence editor is provided in Appendix A. If necessary, use the step by step guide for entering endstate names provided in the tutorial.

   The procedure for entering the endstate name using the sequence editor requires the following:

   a. Entering the Analyze Sequence option from the main menu.

   b. Selecting the Link Event Trees option.

   c. Highlighting the WORK event tree.

   d. Selecting the Sequence Editor.

   e. Using the Line Edit option to enter the endstate name data. An additional discussion of this feature is presented in the IRRAS Reference Manual.[2]

3. Endstate name data can be entered into the event tree flat file (.ETG or .ETL) extracted from the IRRAS program using a text editor. After modification, the file must be loaded as described in Appendix A. This may be the fastest method available but requires more substantial steps and, therefore, is prone to errors. This method is not discussed further.

4.4.2 Entering Endstates for Analysis

Even though the endstate names may appear in the graphics, they will not be available for analysis until each endstate has been added in the Modify Data Base module, or the sequences in the event tree are generated.

Methods for Entering Endstates for Analysis.

1. Endstate data can be entered in the Modify Data Base module. This is perhaps the easiest method as there are usually not a large number of endstates and it does not require leaving the IRRAS environment. Though it may be slower than the other method discussed, it is recommended for most situations. Additional information concerning endstate names is contained in the IRRAS Reference Manual.[2]

   The procedure for entering the endstate data requires the user to (a) enter the Modify Data Base module from the main menu, (b) select the End States option, and (c) add the endstate names in the Edit End States Menu.

2. These data can be entered into the event tree flat file (.ESD) extracted from the IRRAS program, using a text editor. After modification, the file must be loaded as described in Appendix A. This method is not discussed further.

4.4.3 Entering Endstate Description and Text

Descriptions and text associated with event tree endstates can also be entered, though it is unnecessary for analysis.

Methods for Entering Endstate Description and Text.

1. Endstate description and text can be entered in the Modify Data Base module. This is perhaps the easiest method as it does not require leaving the IRRAS environment. Though it may be slower than the other method discussed here, it is recommended for most situations. Additional information concerning description and text is contained in the IRRAS Reference Manual.[2]

   The procedure for entering the endstate description and text requires:

   a. Entering the Modify Data Base module from the main menu

   b. Selecting the End States option

c. Adding the endstate names and descriptions in the Edit End States Menu

d. Selecting the Text option while highlighting the endstate of interest to enter end state text.

2. Endstate description and text can be entered into the endstate description flat file (.ESD) extracted from the IRRAS program, using a text editor. The endstate textual data can be entered into the endstate text flat file (.EST) using the SAPHIRE format. (This is also true of the .ESD). After modification or development, both files must be loaded as described in Appendix A. This method is not discussed further.

4.5 Loading the Fault Tree Data

This section describes loading the database fault trees and associated data and verifying their accuracy. Again, it may be more appropriate to enter data in a different order, depending on the type of data. For nuclear power plant PRAs, the order of data loading presented in this manual has been found to be the most efficient. Fault trees are used in PRAs to represent system failure logic. The sample database has four fault trees, each representing a different top event in the event trees as shown in Figures 3-2 through 3-5.

The IRRAS software contains an option for using the "alpha to graphics" feature to convert the alphanumeric logic structure to a fault tree graphics file (.DLS). The alpha-to-graphics conversion will automatically build the graphical fault tree using the fault tree logic (.FTL). It will recognize and place into the fault tree graphic (1) the fault tree description (as found in the .FTD file), (2) the descriptions of any basic events used in the logic (as found in the .BED file), and (3) all gate descriptions used in the logic (as found in the .GTD file). If, at the time of conversion, this information is not loaded into the interactive database, IRRAS will use default names or blanks. The alpha-to-graphics conversion procedure is provided in Appendix A. The alpha-to-graphics conversion is a very powerful tool but will require some familiarity before it is possible to take full advantage of its usefulness.

Note:

- A .DLS file will be generated during the alpha-to-graphics conversion process and will be located in the family directory.
- Changes to gates and basic events can be made in the MODIFY BASIC EVENTS menu. An alpha-to-graphics conversion performed on the fault tree after the change will implement the change in the graphics.
- Fault tree, basic event, and gate descriptions will not appear in the graphics text boxes (the default is blank) if the appropriate data have not been loaded into the database.

There are four overall methods to develop fault tree graphics that represent logic, depending on whether the data are available electronically or in hardcopy.

1. If hard copy data are available that contain the fault tree structure in graphics form,

   a. Create the fault tree graphics files (.DLS) in the IRRAS-build fault tree module, adding the basic event and gate names.

   b. Extract the necessary flat files to enter the basic event descriptions (.BED—Section 4.6.2), fault tree descriptions (.FTD—Section 4.5.2), and gate descriptions (.GTD—Section 4.5.3);load these modified files and use the alpha-to-graphics conversion option (Appendix A) to enter the data into the graphics.

2. If hard copy data contain the fault tree structures defined as logic,

   a. Use a text editor to enter the logic in the fault tree logic file (.FTL) format.

   b. Use a text editor to develop files that contain the basic event descriptions (.BED - Section 4.6.2), fault tree descriptions (.FTD - Section 4.5.2), and gate descriptions (.GTD - Section 4.5.3) in the correct formats.

   c. Load these files into IRRAS (see Appendix A for the procedure.)

   d. Use the alpha-to-graphics conversion to develop the graphical representation of the fault trees (see Appendix A for procedure).

3. If electronic data contain fault tree logic structures that are compatible with IRRAS, directly load the file into IRRAS.

4. If electronic data contain a fault tree defined as logic that is not compatible with IRRAS,

   a. It may be possible to convert these files into a form that can be entered directly into IRRAS using programming (e.g., BASIC or FORTRAN) or an editing tool with a macro language (e.g., EXCEL or KEDIT). If this is not possible, either use any available hard copy graphics or printout the logic and use the methods discussed above to enter the information.

The following steps must be performed to actually load and verify all the fault tree data.

1. Entering the fault tree logic (section 4.5.1)

2. Entering the fault tree descriptions and text (section 4.5.2)

3. Entering the gate descriptions and attributes (section 4.5.3)

4. Generating system cut sets (section 4.5.4).

4.5.1 Entering Fault Tree Logic

The fault tree data entry is complicated by the fact that SAPHIRE uses an interactive database. Information entered in the process of graphical fault tree construction are used in many areas of the program. Graphical data structure translated into logic and other information are entered into the interactive database using internal lists. Such information includes the type of gates and basic events used, the textual descriptions entered in gate and basic event boxes, and the textual descriptions added for a fault tree description. The information on these internal lists can subsequently be extracted into IRRAS flat files. Conversely, IRRAS can be used to build fault tree graphics from logic and descriptions entered in the database.

> **Note:** When a new fault tree is saved, a .DLS file is automatically created in the family subdirectory. The graphics file is translated into internal fault tree logic. As a result of entering the fault tree graphics, the .FTL, .FTD, .GTA, and .GTD (fault tree logic, fault tree description, fault tree gate attributes, and fault tree gate descriptions, respectively) files can be extracted from the interactive database. SAPHIRE will provide default gate and basic event names. Therefore, it is recommended that both gate names and the basic event names be entered at the time the fault tree is built.

There are different methods to enter fault tree logic, depending on what data type is available. The quickest way is to enter existing files (either graphic or logic) if available and compatible. The second best method is to enter the fault tree structure information into SAPHIRE in the graphics module. It is possible, but may be difficult, to develop logic to enter into a flat file from a graphic and then load this file into SAPHIRE. It is relatively straightforward to enter fault tree logic in the graphics module. The process of entering and saving fault trees is discussed in great detail in the IRRAS Tutorial Manual.[3] The fault tree flat files that contain the graphics and logic information are shown in Table 4-10.

Methods for Entering Fault Tree Logic.

1. If hard copy data are available that contain the fault tree structure in graphics form, then create the fault tree graphics files (.DLS) in the IRRAS-build fault tree module.

   The procedure for entering the data using the software requires:

   a. Entering the Build Fault Tree module from the main menu

   b. Selecting the Build Graphic Trees option

   c. Entering the fault tree structure (including the basic event and gate names) as shown in Figures 3-2 through 3-5 (Section 3) using the graphics editor.

Table 4-10. Extracted fault tree logic and graphic flat files.

File	Extracted file information
.FTL	SAMPLE, ALARM = ALARM OR ALARM-1 ALARM-2 ALM-MECH ALARM-1 OR ALM-SWT ALM-FTS ALARM-2 AND ALM-CPF ALM-BPF ^EOS SAMPLE, PERSONAL = PERSONAL OR SICK SICK-FAM OTHER ^EOS SAMPLE, TRNS-2 = TRNS-2 AND PER-TRNS PUB-TRNS-LAT ^EOS SAMPLE, TRNSPRT = TRNSPRT AND PER-TRNS PUB-TRNS
.DLS	Is not in ASCII format and therefore cannot be viewed or edited.

2. If hard copy data contain the fault tree structures defined as logic, then use a text editor to enter the logic in the fault tree logic file (.FTL) format and load this file into SAPHIRE. This method is not discussed further.

3. If electronic data contain the fault tree structures defined as logic that are compatible with IRRAS, directly load the file into IRRAS.

   The procedure for entering the fault tree logic requires

   a. Entering the Utilities module from the main menu

   b. Selecting the Load MAR-D Data Files option

   c. Selecting the correct option from the DATA SOURCE menu (MAR-D or SETS)

   d. Selecting the type of data to enter following the menu directions.

4. If electronic data contain the fault tree structures defined as logic that are not compatible with IRRAS, then it may be possible to convert these files into a form that can be entered directly into IRRAS using programming (e.g., BASIC or FORTRAN). This requires either editing and/or programming skills that are beyond the scope of this manual. If it is not possible to developed a program to convert the files, it may be possible to use available hard copy graphics or print out logic and use the methods discussed above to enter the data.

Note:

- **IMPORTANT:** The fault tree top gate name *must* be named the same as the fault tree file name. In addition, all top gate names must be eight or less characters due to DOS limitations.

- The .DLS file contains fault tree graphical information. To view and modify a fault tree, the .DLS flat file for that fault tree must be available on the subdirectory. Once the graphics file has been loaded into the interactive database, it is not necessary to have the graphics available (for cutset generation and quantification). The .DLS files can be cleared and extracted using the Extract Graphic Fault Trees option from the Build option (see the IRRAS Reference Manual[2]). Also, .DLS files can be extracted from the database as described in Appendix A.

- When building fault trees, ensure that there are no discontinuities in lines connecting gates, events, and transfers. Discontinuities in these lines will interrupt fault tree logic.

- For ease of document control, include the family name, the fault tree name, the title, and the document-related page number in the graphics.

- If a title is added to the fault tree graphic, it needs to be 16 or greater characters long to work correctly with the interactive database and the alpha-to-graphics conversion option. A shorter title will not be recognized by the software.

- While building large fault trees, save them periodically to prevent loss of data due to a power failure.

- A transfer is usually made to the top gate of another fault tree. However, the user can transfer to a gate in the middle of the fault tree on the same page but not to to a gate in a fault tree on a separate page. Transfers to another page must be to a top gate.

- All fault trees are entered into the interactive database system listing as top gate fault trees. It is up to the user to designate these as sub-trees in the Modify Data Base module. This does not affect use of the analysis except that all trees designated as a top gate will appear in the Fault Tree Analysis menu.

4.5.2 Entering Fault Tree Descriptions and Text

As with event trees, many PRAs will contain descriptions and indepth textual discussion on those fault trees considered important to the analysis. The sample database has both description and text for all the fault trees developed for demonstration. Table 4-11 contains the fault tree descriptions and text extracted.

Table 4-11. Extracted fault tree descriptions and text flat files.

File	Extracted file information	
.FTD	SAMPLE	=
	ALARM	,ALARM CLOCK FAILURE
	PERSONAL	,PERSONAL PROBLEMS
	TRNS-2	,COMMERCIAL TRANSPORTATION FAILS AT A LATER TIME
	TRNSPRT	,PERSONAL AND COMMERCIAL TRANSPORTATION FAIL
.FTT	SAMPLE, ALARM= The ALARM fault tree (Figure 3-2) is a simple representation modeling alarm clock failure. Some common reasons for alarm clock failure include setting the wrong time, mechanical failure, or power failure (either battery or commercial).	

Methods for Entering the Fault Tree Descriptions and Text.

1. Fault tree descriptions and text can be entered in the Modify Data Base module. Using this module is perhaps the easiest method as it does not require leaving the IRRAS environment. Though it may be slower than the other method discussed here (depending on the number of fault trees), it is recommended for most situations. Use of the IRRAS text editor in the Text option is described in the IRRAS Reference Manual.[2]

   The procedure for entering the fault tree descriptions and text requires:

   a. Entering the Modify Data Base module from the main menu

   b. Selecting the Systems option

   c. Adding the fault tree descriptions in the Edit Systems Menu

   d. Selecting the Text option while highlighting the fault tree of interest to enter fault tree text.

2. Fault tree descriptions and text can also be entered into the fault tree flat file (.FTD) extracted from the IRRAS program using a text editor. The fault tree textual data can be entered into the fault tree text flat file (.FTT) using the SAPHIRE format. (This is also true of the .FTD). After modification or development both files must be loaded as described in Appendix A. This method is not discussed further.

4.5.3 Entering Gate Descriptions and Attributes

Gate descriptions are usually available in PRAs. They are useful and necessary for clarifying how the system logic was developed for use in future analysis. For example, gate descriptions may designate where certain train logic begins in the fault tree logic so that the branch can be eliminated for analysis.

In the sample database, descriptions are available even though they do not provide any additional information concerning the analysis. Note that the SAPHIRE attribute is the type of gate, (i.e., OR, AND, and TRANSFER). The gate name and this information should already be present in the Modify Data Base module under Gates from entering the fault tree logic into the interactive database. Table 4-12 shows the fault tree gate files extracted.

Methods for Entering Gate Descriptions and Attributes.

1. Gate descriptions and attributes are easily entered into the graphics editor. This method is potentially the most time consuming but the most straightforward. In this case, the fault tree could be finalized and files extracted without any intermediate steps. Both the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] contain details concerning this process.

   The procedure for entering the data in the graphics menu requires the user to (a) enter the Build Graphic Trees option from the main menu, (b) access the Build Graphic menu, and (c) use the Write Text option in the TEXT popup menu to enter the gate description.

2. Gate descriptions and attributes can be entered using the Modify Data Base option and then performing an alpha-to-graphics conversion to place the description in the graphics. This is perhaps the easiest method as it does not require leaving the IRRAS environment. Though it may be slower than the third method discussed here, it is recommended for most situations. See the IRRAS Reference Manual[2] for additional information.

Table 4-12. Extracted fault tree gate flat files.

File	Extracted file information	
.GTD	SAMPLE	=
	ALARM	, ALARM CLOCK FAILURE
	ALARM-1	, ALARM CLOCK SETTING FAILURE
	ALARM-2	, ALARM CLOCK POWER FAILURE
	PERSONAL	, PERSONAL PROBLEMS
	TRNS-2	, COMMERCIAL TRANSPORTATION FAILS AT A LATER TIME
	TRNSPRT	, PERSONAL AND COMMERCIAL TRANSPORTATION FAILURE
.GTA	SAMPLE	=
	* Name	, Type
	ALARM	, OR
	ALARM-1	, OR
	ALARM-2	, AND
	PERSONAL	, OR
	TRNS-2	, AND
	TRNSPRT	, AND

The procedure for entering the gate descriptions and attributes includes the following:

a. Entering the Modify Data Base module from the main menu.

b. Selecting the Gates option.

c. Highlighting the gate name

d. Selecting the Modify option.

e. Using the Modify Gate menu to enter the gate description. When data entry is finalized, perform an alpha to graphic conversion (see Appendix A) to enter this information into the fault tree graphics.

3. Gate descriptions and attributes can be entered using a text editor into the gate description flat file (.GTD) that was extracted from the IRRAS program. After modification, the file must be loaded as described in Appendix A. The attribute file data will have been entered in the process of entering the fault tree logic. It may be useful to extract the gate attribute flat file (.GTA) for some other purpose. This method is not discussed further.

4.5.4 Generating System Cut Sets

It has been noted that some PRAs provide indepth system cutset information while others do not. Having the original system cutsets is very helpful in verifying that the correct logic has been entered into the database. Since most PRAs comprised large system fault trees, it is possible to generate many more cutsets than what may have been reported. In these cases, to duplicate the PRA system cutsets, it may be necessary to vary the probability cutoff used to generate them. Also, for certain databases, it may be impossible to match the system cutsets that are reported in the PRA with those generated in SAPHIRE. This can be due to many reasons, one of which is poor documentation for the original analysis performed. In this case, it may be necessary to manually enter the cutsets into the database. For the sample database, the system cutsets were presented in Section 3. It is important to note that for cut set generation and quantification, SAPHIRE uses only the logic and not the graphical representation of the fault tree. The graphics are useful for easy visualization of the system. Table 4-13 shows the system cut set flat files extracted.

Methods for Generating System Cut Sets.

1. In the process of generating system cutsets, fault tree logic can be verified. The IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] provide additional information on this process.

   The procedure for generating system cutsets and obtaining a report for verification requires the following:

   a. Entering the Fault Tree Analysis module from the main menu.

Table 4-13. Extracted system cut sets flat files.

File	Extracted file information
.FTC	SAMPLE, ALARM, 0001= ALM-FTS + ALM-MECH + ALM-SWT + ALM-BPF * ALM-CPF . ^EOS SAMPLE, PERSONAL, 0001= OTHER + SICK + SICK-FAM . ^EOS SAMPLE, TRNS-2, 0001= PER-TRNS * PUB-TRNS-LAT . ^EOS SAMPLE, TRNSPRT, 0001= PER-TRNS * PUB-TRNS .

b. Selecting the Analyze Systems option.

c. Highlighting the fault tree to analyze and select the Generate Cut Sets option. (When Generate Cut Sets is chosen, an intermediate screen will appear that queries Cut Set Generation Cutoff Values. This is where the probability cutoff can be set to limit the cutsets produced, or can be varied to duplicate the original PRA. See the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] for a discussion of these features.)

d. Pressing enter after selecting the appropriate cutoff values.

e. Entering Display Results from the Fault Tree Analysis menu.

f. Highlighting a fault tree and selecting Cutsets to view cutsets and to produce a report.

2. Cutset data can be entered by first using a text editor to edit the fault tree cutset flat file (.FTC) developed using the SAPHIRE format. After development, the file must be loaded as described in Appendix A. This would only be used in a case where it is impossible to match the database files with the generated cutsets. (This may occur even when the fault tree graphics appear identical.) This is a slower method, and because it requires more steps in the data entry process may be prone to errors. This method is not discussed further.

4.5.5 Verifying the Fault Tree Data

After the logic and data for each fault tree are entered, it is a necessary step to verify that the information entered into the database is correct before proceeding. The recommended method to check the fault tree data is to extract those flat files, reports, and graphics that are the most similar to what is presented in the database and verify their accuracy.

4.6 Loading Basic Event Data

This section discusses loading the basic event information such as probabilities, calculation types, and attributes. As event tree files (see Section 4.2) and fault tree files (see Section 4.5) are created or loaded, SAPHIRE constructs an internal list of all basic events, undeveloped events, gates, initiating events, and top events. These are added to the interactive database Basic Event list found in the Modify Data Base module. Because this list will not be complete, the user will still need to enter probability values, descriptions, and other detailed information, as necessary. Additionally, new basic events may need to be added to account for beta factors, recovery actions, and other factors. For more information on IRRAS operation as it relates to basic event information, consult the tutorial and technical reference manuals.

SAPHIRE offers a wide range of options for entering basic event information. It is not possible in this manual to completely address all the combinations available for basic event data. This section discusses many of these basic event data features and presents the methods for entering these data. The Modify Event menu is shown in Figure 4-3. This menu contains seven areas for data entry. A brief discussion of the features available in the basic event data screen follows.

EVENT DATA

Primary Name — The primary event name is 16 characters in length and is user-defined.

Alternate Name — The default is the same as the primary name. Analysis can be done using either the primary or alternate names. This allows the user to have two naming schemes in use.

Group Name — This is used in conjunction with the P&ID editor. It allows the grouping of several failure events around a component; for example, a pump may fail to start, fail to run, and fail due to test or maintenance. Using the group name feature, these three failure events can be brought to view from the P&ID editor. The group is also used for the "propagate event failures" option.

Process Flag — A process flag is a one-character field that specifies if certain processes should take special note of this event. It allows a basic event to be flagged for sensitivity analysis or as a zone event. It allows sequence top events to use combinations of system logic or developed events for both success and failed logic. It allows the system logic to be treated as a single value developed event (i.e., split fraction) instead of using the entire system fault tree logic. For more information on process flags see the SAPHIRE Technical Reference Manual.[4]

```
┌──── Family ────┐     ┌─────────────────────┐
│ SAMPLE         │     │ M o d i f y  E v e n t │
└────────────────┘     └─────────────────────┘

                  Option |M|   Exit / Modify

┌───────────────────────────── Event Data ──────────────────────────────┐
│ Names Primary                    Process      Category      Component │
│     Alternate                    Flag                       Id        │
│         Group                                                         │
│   Description                                                         │
├──── Failure Data ──────┬────── Seismic Fragility ─────┬── Attributes ──┤
│ Calculation Type       │ Type   Name                  │ System         │
│ Probability  ------E----│ Beta Random       ------E----│ Train          │
│ Lambda       ------E----│ Beta Uncertainty ------E----│ Type           │
│ Tau          ------E----│ F. Acceleration  ------E----│ Failure Mode   │
│ Mission Time ------E----│ Correlation Class           │ Location       │
├─── Uncertainty Data ───┼──────────────── Transformations ──────────────┤
│ Distribution Type      │          Type               Level  ---        │
│ Name                   ├──────────────── Susceptibilities ─────────────┤
│ Value 1      ------E----│ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 │
│ Value 2      ------E----│ N  N  N  N  N  N  N  N  N  N  N  N  N  N  N  N │
│ Correlation Class      │                                                │
└──────────────── Press <F1> for Help or List of Values ─────────────────┘
```

Figure 4-3. The IRRAS modify event menu.

Category — A single character field specifies the category or use of an event:

- ' ' General purpose event
- 'I' Initiating Event
- 'H' Hazard event (system specified only)
- 'R' Recovery event
- 'S' Support system state

Component ID — Component identifier is a user-defined, seven-character, uppercase alphanumeric field that identifies a component by a unique designator, typically part of the component label.

FAILURE DATA

Calculation Type — The failure data calculation type for each basic event can be entered in the calculation type field. The SAPHIRE menu showing the options available is presented in Figure 4-4. The calculation type selected will determine whether data needs to be entered in the probability, lambda, tau, and mission time areas. When adding basic events in SAPHIRE, the default for the basic event calculation type is 1. The SAPHIRE Technical Reference Manual[4] contains more details on each of the specific calculation types.

```
Failure Data Calculation Types

1 - Probability
2 - Lambda * Mission Time
3 - 1 - Exp(-Lambda * Mission Time)
4 - Lambda * Min(Mission Time, Tau)
5 - Operating Component with Repair (Full Eq)
6 - Lambda * Tau / 2.0
7 - 1 + (EXP(-Lambda*Tau)-1.0) / (Lambda*Tau)
8 - Base Probability + Probability Field Value
9 - Base Probability * Probability Field Value
T - Set to House Event (Failed, Prob=1.0)
F - Set to House Event (Successful, Prob=0.0)
I - Ignore this Event (Remove it from logic)
S - Set to System Min Cut Upper Bound
G - Use Probability for Ground Acceleration
L - Low Hazard Curve
M - Medium Hazard Curve
H - High Hazard Curve

Calculation Type ====>1
```

Figure 4-4. SAPHIRE calculation type menu.

UNCERTAINTY DATA

Distribution Types — The SAPHIRE menu showing the uncertainty distribution types that can be entered are shown in Figure 4-5. The value entered here will dictate what will appear in the name area and what input is necessary in Value 1.

Correlation class — The correlation class is a four-character identifier specified by the user to identify data dependencies among like events. A correlation class of 0 or a space indicates that there are no data dependencies. The user should assign like events the same correlation class identifier. The identifier can be a combination of letters and numbers. During uncertainty calculations, the same sample value will be used for all events that have the same correlation class identifier. The default is a space (i.e., no correlation).

Type	Distribution Values
L	Log Normal, Error Factor
N	Normal, Standard Dev.
B	Beta, b of Beta(a,b)
G	Gamma, a of Gamma(a)
C	Chi-Squared, Degrees of Freedom
E	Exponential, none
U	Uniform, Upper End Pt.
H	Histogram, Histogram Number

Uncertainty Distribution Types

Distribution Type═══>L

Figure 4-5. SAPHIRE uncertainty distribution type menu.

ATTRIBUTES

System — The system code allows the user to assign a system name to events in the same system. This code can later be used for various sorting and analysis processes available in SAPHIRE.

Train — The train code allows the user to assign a train name to events in the same train. This code can be used later for various sorting and analysis processes available in SAPHIRE.

Type — The type code allows the user to assign a type name to events of the same type. This code can be used later for various sorting and analysis processes available in SAPHIRE.

Failure Mode — The failure mode code allows the user to assign a failure mode to events with the same failure mode. This code can be used later for various sorting and analysis processes available in SAPHIRE.

Location — The location code allows the user to assign a location to events located in the same area. This code can be used later for various sorting and analysis processes available in SAPHIRE.

The following steps must be performed to actually load and verify all the basic event data:

1. Adding basic events (Section 4.6.1)

2. Adding basic event descriptions (Section 4.6.2)

3. Entering basic event data (availability and uncertainty) (Section 4.6.3).

4.6.1 Adding Basic Events

Basic events not listed in either the fault trees or event trees may be necessary in a PRA to accommodate special situations such as substitutions or recovery actions. The sample database requires the entry of one recovery action basic event. This is shown in the basic event listing in Section 3. (Note that SAPHIRE will also allow the user to enter a new basic event in the Recover cutset module.)

Methods for Adding Basic Events.

1. Basic events can be entered in the Modify Data Base module. Using this method is perhaps the easiest because it does not require leaving the IRRAS environment. Though it may be slower than the other method discussed here, it is recommended for most situations. See the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] for more information.

   The procedure for entering the basic event requires the user to (a) enter the Modify Data Base module from the main menu, (b) select the Basic Event option, and (c) select the Add option.

2. Basic events also can be entered using a text editor by modifying the basic event flat file (.BED) that was extracted from the IRRAS program. After modification, the file must be loaded as described in Appendix A. This may be the fastest method available but requires more substantial steps and may be prone to errors. This method is not discussed further.

4.6.2 Adding Basic Event Descriptions

Basic event descriptions are commonly used in PRAs. When entered into the interactive database, the alpha-to-graphics conversion can be used to place the descriptions into the fault tree graphics. Table 4-14 shows the basic event description flat file extracted from the sample database.

Methods for Adding Basic Event Descriptions.

1. Basic event descriptions can be entered in the Modify Data Base module. This method is perhaps the easiest because it does not require leaving the IRRAS environment, but it is not generally recommended for most databases since the number of basic events is large. See the IRRAS Reference Manual[2] for more information.

   The procedure for entering the basic event descriptions requires the user to (a) enter the Modify Data Base module from the main menu, (b) select the Basic Event option, and (c) select the Modify option.

2. Basic event descriptions can be entered using a text editor by modifying the basic event flat file (.BED) that was extracted from the IRRAS program. After modification, the file must be loaded as described in Appendix A. This is the fastest method available and, due to the large number of basic events common in most PRAs, it is recommended over method A. This method is not discussed further.

Table 4-14. Extracted basic event descriptions flat file.

File	Extracted file information	
.BED	SAMPLE	=
	<FALSE>	,System Generated Success Event
	<PASS>	,System Generated Ignore Event
	<TRUE>	,System Generated Failure Event
	ALARM	,Alarm system fault tree
	ALM-BPF	,Alarm fails due to battery failure
	ALM-CPF	,Alarm fails due to commercial power failure
	ALM-FTS	,Alarm fails because worker fails to set
	ALM-MECH	,Alarm fails due to mechanical failure
	ALM-SWT	,Alarm fails because worker set wrong time
	MEDICINE	,Recovery for sick failure preventing attending work
	OTHER	,Other personal reasons that cause a failure to get to work
	PER-TRNS	,Personal transportation
	PERSONAL	,Personal reasons for failure system fault tree
	PUB-TRNS	,Public transportation fails
	PUB-TRNS-LAT	,Public transportation fails late time frame
	SICK	,Failed to get to work because of illness
	SICK-FAM	,Failed to get to work because of illness in family
	TRNS-2	,Transportation system fault tree-late time frame
	TRNSPRT	,Transportation system fault tree
	WORK	,Event tree (WORK) initiating event

4.6.3 Entering Basic Event Data

To determine the frequency of failure in a SAPHIRE analysis, it is necessary to enter the probability or frequency of failure for each basic event. Most PRAs may have several calculation types, the most common being failure on demand, failure over a mission time, and standby failure rates. Also, PRAs generally address uncertainty and will provide either a standard deviation or error factor to be entered in the analysis. It is beyond the scope of this document to present all the possible applications available. The SAPHIRE Technical Reference Manual[4] provides a detailed discussion on many of the features available. The sample database contains limited examples and is presented for illustration only. Table 4-15 shows the basic event data flat files extracted from the sample database.

Methods for Entering Basic Event Data.

1. Basic event data can be entered in the Modify Data Base module. Using this module is perhaps the most straight forward method as it does not require leaving the IRRAS environment. But, in this case it is not the recommended method because of the many keystrokes that are necessary. See the IRRAS Reference Manual[2] for more information.

Table 4-15. Extracted basic event data flat files.

File	Extracted file information
.BEA	SAMPLE =
	* Name ,AltName ,Typ,Sys,Fail,Loc,CompId, GroupName ,Train,Attributes
	<FALSE> ,<FALSE> , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	<PASS> ,<PASS> , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	<TRUE> ,<TRUE> , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALARM ,ALARM ,DE , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALM-BPF ,ALM-BPF , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALM-CPF ,ALM-CPF , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALM-FTS ,ALM-FTS , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALM-MECH ,ALM-MECH , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	ALM-SWT ,ALM-SWT , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	MEDICINE ,MEDICING , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	OTHER ,OTHER , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	PER-TRNS ,PER-TRNS , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	PERSONAL ,PERSONAL ,DE , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	PUB-TRNS ,PUB-TRNS , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	PUB-TRNS-LAT ,PUB-TRNS-LAT, , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	SICK ,SICK , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	SICK-FAM ,SICK-FAM , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	TRNS-2 ,TRNS-2 ,DE , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	TRNSPRT ,TRNSPRT ,DE , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
	WORK ,WORK , , , , , , , ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
.BEI	SAMPLE =
	* Name ,FdT,UdC ,UdT,UdValue, Prob , Lambda , Tau , Mission ,Init
	<FALSE> ,1, ,L, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	<PASS> ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	<TRUE> ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALARM ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALM-BPF ,1, ,L, 3.000E+000, 9.000E-008,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALM-CPF ,1, ,L, 3.000E+000, 1.500E-002,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALM-FTS ,1, ,L, 1.000E+001, 5.500E-006,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALM-MECH ,1, ,L, 3.000E+000, 2.700E-008,+0.000E+000,+0.000E+000,+0.000E+000, ,
	ALM-SWT ,1, ,L, 1.000E+001, 2.700E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	MEDICINE ,1, ,L, 5.000E+000, 5.000E-001,+0.000E+000,+0.000E+000,+0.000E+000,R,
	OTHER ,1, ,L, 1.000E+001, 8.100E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	PER-TRNS ,1, ,L, 5.000E+000, 5.500E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	PERSONAL ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	PUB-TRNS ,1, ,L, 3.000E+000, 2.700E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	PUB-TRNS-LAT ,1, ,L, 3.000E+000, 2.000E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	SICK ,1, ,L, 1.000E+001, 8.100E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	SICK-FAM ,1, ,L, 1.000E+001, 4.000E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
	TRNS-2 ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	TRNSPRT ,1, ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
	WORK ,1, ,L, 2.000E+000, 2.480E+002,+0.000E+000,+0.000E+000,+0.000E+000,I,

The procedure for entering the basic event data requires

a. Entering the Modify Data Base module from the main menu

b. Selecting the Basic Event option

c. Highlighting the basic event and selecting Modify in the Edit Events menu

d. Entering the basic event values and pressing <Enter> to save.

2. Basic event data also can be entered using a text editor by modifying the basic event information flat file (.BEI) that was extracted from the IRRAS program. After modification, the file must be loaded as described in Appendix A. Using this technique is the recommended method (though the file will need to be reloaded after modification) since it requires substantially fewer keystrokes and is the fastest method available. This method is not discussed further.

Note:

- Not all the information for a basic event needs to be entered for calculation purposes. The information required is the primary name, the category, the calculation type, the probability value, and the uncertainty distribution type and value (uncertainty is only necessary if an uncertainty calculation is to be performed).

- When a basic event is added to the IRRAS internal list, it is assigned default values for uncertainty and failure data.

- Important — remember to generate event data as described in Appendix A to obtain an updated current case database. Any basic event values input into the Modify Data Base module will appear only in the base case data (which is not used for analysis) until this procedure has been performed. See the discussion on the base and current case Section 2.3.

4.7 Loading Sequence Data

This section discusses the loading of sequence data, including cutsets, text, and descriptions. Sequences are used in PRAs to develop the overall CDF value and to identify those scenarios of events that are of concern to plant safety. Sequences with similar outcomes are grouped by endstates for evaluation in the level 2 and 3 analysis. Most PRAs present the dominant (or greatest contributors to CDF) sequence cutsets.

The following steps must be performed to actually load and verify all the sequence data:

1. Generating sequence cutsets (Section 4.7.1)

2. Entering the sequence description and text (Section 4.7.2).

4.7.1 Generating Sequences Cutsets

Since some PRAs have event trees that link to large system fault trees, it is possible to generate a large number of cutsets. The probability cutoff option and the size cutoff limits the number of cutsets to those above a certain value and order. This cutoff can be manipulated so that the cutsets match those produced by the PRA. For certain databases, it may be impossible to match the sequence cutsets that are reported in the PRA with those generated by SAPHIRE. This difference can be due to many reasons, one of which is poor documentation for the original analysis performed. In this case, it may be necessary to manually enter the cutsets into the database.

The sequence cutsets for the sample database are reported in Section 3. There was no cutoff used for this very simple problem. It is important to note that for cut set generation and quantification, SAPHIRE uses only the logic and not the graphical representation of the fault tree. Table 4-16 shows the sample database sequence cutsets.

Methods for Generating Sequence Cutsets.

1. To have SAPHIRE generate sequence cutsets, use the Analyze Sequences module. The IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] provide additional information on this process.

   The procedure for generating sequence cutsets requires the following:

   a. Entering Analyze Sequences from the main menu.

   b. Selecting the Analyze Sequences option.

   c. Highlighting the sequence to analyze and selecting the Generate Cut Sets option. (When Generate Cut Sets is chosen, an intermediate screen will appear that queries Cut Set Generation Cutoff Values. This is where the probability cutoff can be changed to limit the cutsets produced or can be varied to duplicate the original PRA. See the IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] for a discussion of these features.

Table 4-16. Extracted sequence cut sets flat files.

File	Extracted file information
.SQC	SAMPLE, WORK, 2, 0001= PER-TRNS * PUB-TRNS . ^EOS SAMPLE, WORK, 3, 0001= OTHER + SICK * MEDICINE + SICK-FAM . ^EOS SAMPLE, WORK, 4, 0001= ALM-FTS + ALM-MECH + ALM-SWT + ALM-BPF * ALM-CPF . ^EOS SAMPLE, WORK, 5, 0001= ALM-FTS * PER-TRNS * PUB-TRNS-LAT + ALM-MECH * PER-TRNS * PUB-TRNS-LAT + ALM-SWT * PER-TRNS * PUB-TRNS-LAT + ALM-BPF * ALM-CPF * PER-TRNS * PUB-TRNS-LAT .

d. Pressing enter after selecting the appropriate cutoff values.

e. Entering Display Results from the Event Tree Analysis menu.

f. Highlighting the event tree and selecting Cutsets to view cutsets and produce a report on the screen or printer.

2. Using a text editor, cutset data can be entered into a sequence cutset flat file (.SQC) format. After development, the file must be loaded as described in Appendix A. This would only be used in a case where it is impossible to match the database files with the generated cutsets. (This may occur even when the logic appears identical.) This method is not discussed further.

4.7.2 Entering the Sequence Description and Text

It is common that PRAs will discuss in detail the dominant sequences that were identified. The accident scenarios and recovery actions applied may be described in detail. The sample database contains a brief description and some text information for the sequences. Table 4-17 shows the sample database sequence description and text flat files.

Methods for Entering the Sequence Description and Text.

1. The sequence description and text can be entered in the Modify Data Base module. This technique is perhaps the easiest method as it does not require leaving the IRRAS environment. Although it may be slower than the other method discussed below, it is recommended for most situations. Additional information concerning adding descriptions and text is contained in the IRRAS Reference Manual.[2]

   The procedure for entering the sequence description and text requires

   a. Entering the Modify Data Base module from the main menu

   b. Selecting the Event Tree option

Table 4-17. Extracted sequence description and text flat files.

File	Extracted file information
.SQD	SAMPLE, WORK= 2 ,LATE-TO WORK 3 ,MISS-WORK 4 ,LATE-TO-WORK 5 ,LATE-TO-WORK
.SQT	SAMPLE, WORK, 3= Sequence 3 is the event tree sequence used to demonstrate the use of recovery rules or recovery actions.

c. Selecting Sequences from the Edit Event Trees menu (note that sequences will have to be generated as described in Section 4.7.1, before appearing in this menu)

d. Highlighting the sequence and selecting the Modify option to add a sequence description in the Edit Sequences menu

e. Highlighting the sequence and selecting the Text option to add text in the Edit Sequence menu.

2. Using a text editor, the sequence description and text can be entered into the sequence description flat file (.SQD) format. The sequence textual data can be entered into the sequence text flat file (.SQT) using the SAPHIRE format. (This is also true of the .SQD). After modification or development both files must be loaded as described in Appendix A. This method is not discussed further.

4.8 Recovery Actions

This section discusses the addition of recovery actions to sequence cutsets. PRAs often have recovery actions applied to a specific scenario of events that may occur in a sequence or fault tree cutset. These recovery actions are not directly modeled in either an event tree or fault tree and may be required to be added to the cutsets to obtain a result comparable to the PRA. The sample database has a very simple recovery action that will be applied to one sequence cutset. Recovery actions or recovery rules can be applied to fault tree cut sets using System and System-Family Rules. Recovery actions can also be applied to event tree sequence cut sets by using Family, Event Tree, and Sequence Rules. Both are used in a similar method. An example of a Family Rule recovery action being applied to sequence cut sets would be the case of double maintenance events not allowed by technical specifications. The sample database contains a simple example of a recovery action applied to a sequence cutset.

METHOD

1. This method discusses how to use SAPHIRE to apply recovery actions from the Analyze Sequences module. The method will apply recovery actions to sequence cut sets, but system cut set recovery actions are similar. The IRRAS Reference Manual[2] and the IRRAS Tutorial Manual[3] provide additional information on this process.

   The procedure for applying recovery requires the following:

   a. Selecting Analyze Sequences from the IRRAS main menu.

   b. Selecting the Recover Cut Sets option from the Event Tree Analysis menu.

   c. Highlighting a sequence and selecting Sequence Rules from the Recovery menu. (In this case recovery actions could be applied to the sequence cut sets by Family, Event Tree, or Sequence Rules. Since there is only one event tree, and it is in the family, the Sequence Rules will be used.)

   d. Selecting the Apply or Edit Rules option from the Recovery menu.

e. Typing <F9> to Insert Rule, bringing up the Recovery logic menu.

f. Moving to the square under Match and typing A. (This will add recovery actions to all the sequences that match the recovery criteria.)

g. Moving to the top of the left-most Events column, typing SICK.
(The NOT column (Type N) is used to indicate the events to which the recovery actions are *NOT* applied. This can be used to eliminate certain combinations of events to which recovery will not be applied.)

h. Moving to the left-most Recovery Events block and typing MEDICINE. (This is the recovery action that will be applied to all cut sets that contain the basic event SICK in sequence 3).

i. Pressing <Enter>.
(The user can enter a suitable description or, if left blank, IRRAS will fill in the line with event names entered.)

j. Pressing ESC twice and entering Y to save changes.

k. Highlighting sequence 3 and selecting the Apply Rules option from the main Recovery menu. There are four menus from which recovery can be applied. There are embedded Recovery menus in the Sequence Rules option, the Event Tree Rules option, and the Family Rules option. Each of these embedded menus can be used to apply recovery specifically to a sequence, event trees or the family only. The main Recovery screen can be used to apply any combination of the three rule types (Family, Event Tree, or Sequence).

l. Selecting <Y> for Sequences.

The recovery action MEDICINE can be viewed in sequence 3 in the Display Results option under the Event Tree Analysis menu.

4.9 Generating Uncertainty (This section will discuss the generation of system cut set uncertainty.)

Uncertainty of the cutset and endstate results is commonly reported in the PRAs. Both a Monte Carlo and Latin Hypercube option are available in SAPHIRE. It is sometimes difficult to compare SAPHIRE results with those reported in a PRA, because there will be an expected variability between the uncertainty runs depending on the algorithms used, the number of samples, and the seed numbers chosen. It is sometimes necessary to validate these values using hand calculations of the confidence intervals for certain databases.

The following steps must be performed to generate an uncertainty analysis for the database and verify it against the PRA:

1. Generating uncertainty for system cutsets (Section 4.9.1)

2. Generating uncertainty for sequence cutsets (Section 4.9.2)

3. Generating uncertainty for endstates (Section 4.9.3)

4. Generating uncertainty for groups of sequences or the family (Section 4.9.4).

4.9.1 Generating Uncertainty for System Cutsets

It is not unusual to find that a system uncertainty analysis was reported for those PRAs that provided system cutsets. The sample database provides the results to an uncertainty analysis. There are no flat files that can be extracted for system uncertainty. Limited reports can be produced in the Display Results option in the Analyze Fault Trees module.

METHOD

1. Uncertainty can only be produced after cutsets have been generated. Further discussions on uncertainty analysis are found in both the reference and technical manuals.

   The procedure for generating system uncertainty requires the following

   a. Entering the Fault Tree Analysis module from the main menu.

   b. Selecting the Analyze Systems option

   c. Highlighting the system and selecting the Uncertainty option in the Analyze Systems menu

   d. Selecting the uncertainty types and values to use in the Uncertainty Calculation Values menu, and press <enter>.

   e. Viewing the Uncertainty Results on the temporary screen. (A printout of this screen is sometimes useful. It is necessary to either to send a copy to the screen.cpy file on the family subdirectory by pressing <CTRL K> or to produce a screen copy.)

   f. To view the uncertainty stored in the database, enter the Display System Results from the Analyze Systems menu.

   g. Highlighting the system and selecting the Uncertainty option will bring up the Uncertainty Data screen. (Data will be contained for either the current case or both the current and base case depending on whether a base case update has been performed. See Section 2.3 for a discussion of the basecase update feature.)

4.9.2 Generating Uncertainty for Sequence Cutsets

Most PRAs provide sequence cutset uncertainty. Again, it may be difficult to compare SAPHIRE results with those reported in a PRA because there will be an expected variability between the uncertainty runs, depending on the algorithms used, the number of samples, and the seed numbers chosen. The sample database provides the seed number and was developed on SAPHIRE using the Monte Carlo algorithm and, therefore, it should be possible to produce the same results.

METHOD

1. Uncertainty can only be produced after cutsets have been generated and there is only one method to generate it. Further discussions on uncertainty analysis are found in both the IRRAS Reference Manual[2] and the SAPHIRE Technical Reference Manual.[4]

   The procedure for generating sequence uncertainty requires the following:

   a. Entering the Analyze Sequences module from the main menu.

   b. Selecting the Analyze Sequences option.

   c. Highlighting the sequence and selecting the Uncertainty option in the Analyze Sequences menu.

   d. Selecting the uncertainty types and values to use in the Uncertainty Calculation Values menu, and press <enter>. (If several sequences are marked for analysis, the Uncertainty Calculation Values menu will query to select either a group or single analysis.)

   e. View the Uncertainty Results on the temporary screen. (A printout of this screen is sometimes useful. It is necessary to either to send a copy to the screen.cpy file on the family subdirectory by pressing <CTRL K> or to produce a screen copy.)

To view the uncertainty stored in the database, enter the Display Sequence Results from the Analyze Sequences menu. Highlighting the sequence and selecting the Uncertainty option will bring up the Uncertainty Data screen. (Data will be contained for either the current case or both the current and base case depending on whether a base case update has been performed. See Section 2.3 for a discussion of the basecase update feature.)

4.9.3 Generating Uncertainty for Endstates

Very few PRAs provide endstate uncertainty. Again, it may be difficult to compare SAPHIRE results with those reported in a PRA because there will be an expected variability between the uncertainty runs, depending on the algorithms used, the number of samples, and the seed numbers chosen. The sample database provides the seed number and was developed on SAPHIRE using the Monte Carlo algorithm and, therefore, it should be possible to reproduce the uncertainty results. There is no flat file currently available for this option.

METHOD

1. Uncertainty can only be produced after sequence and endstate cutsets have been generated. Further discussions on uncertainty analysis are found in both the IRRAS Reference Manual[2] and SAPHIRE Technical Reference Manual.[4]

The procedure for generating endstate uncertainty requires

a. Entering the Endstate Analysis module from the main menu

b. Selecting the Analyze End States option

c. Selecting the Generate Cut Sets in the Analyze End States menu

d. Highlighting the endstate and generate cutsets

e. Selecting the End State Uncertainty option in the Analyze Endstate menu

f. Highlighting the endstate and selecting the Uncertainty Analysis in the Analyze End States menu.

g. Selecting the uncertainty types and values to use in the Uncertainty Calculation Values menu, and press <enter>.

h. Viewing the Uncertainty Results on the temporary screen (a printout of this screen is sometimes useful.)

To view the uncertainty stored in the database, enter the Display Result from the Endstate Analysis menu. Highlight the endstate and select the Uncertainty option to bring up the Uncertainty Data screen. (Data will be contained for either the current case or both the current and base case, depending on whether a base case update has been performed. See Section 2.3 for a discussion of the basecase update feature.)

4.9.4 Generating Uncertainty for Groups of Sequences or the Family

Most PRAs provide sequence uncertainty, but only a few may perform uncertainties on groups of sequences that are not grouped previously by endstate. Also, some PRAs provide the results of a family uncertainty. The procedure is the same to generate either groups or family uncertainty and, therefore, is presented together. It may be difficult to compare SAPHIRE results with those reported in a PRA because there will be an expected variability between the uncertainty runs, depending on the algorithms used, the number of samples, and the seed numbers chosen. The sample database provides the seed number and was developed on SAPHIRE using the Monte Carlo algorithm and, therefore, it should be possible to produce the same results.

METHOD

1. Uncertainty can only be produced after sequence cutsets have been generated and there is only one method to generate it. Further discussions on uncertainty analysis are found in both the IRRAS Reference Manual[2] and the SAPHIRE Technical Reference Manual.[4]

   The procedure for generating sequence group or family uncertainty requires the following:

   a. Entering the Analyze Sequences module from the main menu.

b. Selecting the Analyze Sequences option.

c. Highlighting a group or the family and selecting the Uncertainty option in the Analyze Sequences menu.

If sequences are to be grouped for analysis, highlight each sequence and use the <F2> key to mark with an asterisk each sequence in the group. If a family uncertainty is to be performed use the <F3> key to mark with an asterisk all sequences. When all the sequences are marked, the Uncertainty Calculation Values Screen will contain the Family uncertainty option.

d. Selecting the uncertainty types and values to use in the Uncertainty Calculation Values menu, and pressing <enter>. If all sequences have been marked, the Uncertainty Calculation Values menu will query to select either a group, single, or family analysis. If only a group of sequences have been marked, the Uncertainty Calculation Values menu will query to select either a group or single analysis.

e. Viewing the Uncertainty Results on the temporary screen. (A printout of this screen is the only result that is available and will need to be produced.)

No uncertainty values are stored in the database for groups of sequences.

4.10 Additional Features

This section discusses additional features that may be necessary in the data loading process, such as house events, change sets, mutually exclusive events, process flags, and importance measures.

The features in this section have proved useful for manipulation of the PRA databases. The sample database is limited in the amount of additional features that can be demonstrated in it while maintaining its simplicity. The features discussed briefly in this section include

1. Use of change sets (Section 4.10.1)
2. Use of house events (Section 4.10.2)
3. Use of process flags (Section 4.10.3)
4. Use of mutually exclusive top events (Section 4.10.4)
5. Use of flag sets (Section 4.10.5)
6. Use of importance measures (Section 4.10.6).

4.10.1 Use of Change Sets

Change sets are used to modify the current case basic event data to accommodate special situations (such as sensitivity analysis) in the data analysis. Modifications made possible by change sets include individual probability changes to a basic event and class probability changes to a group of basic events. A number of different change sets can be added to a database and many combinations of change sets can be implemented. These change sets, containing information about the probability/class changes, can be applied to basic events during system or sequence analysis. Change set modifications are used most often for setting house events with a calculation type F or T in the FAILURE DATA field and PROCESS FLAGS X and Y. A detailed description and example of using a change set is provided in Appendix A.

4.10.2 Use of House Events

A house event is useful in turning on and off sections of a fault tree. For example, often a system is modeled with AC power available. Given that a PRA is modeling a scenario where the offsite power had failed, then sections of the system may become unavailable for accident mitigation. A house event can be used to turn on and off those applicable sections of a system fault tree to provide the correct model. See the SAPHIRE Technical Reference Manual[4] for a detailed description of how house events are used in SAPHIRE.

4.10.3 Use of Process Flags

The use of process flags allows the analyst to manipulate the evaluation of success and failure logic in the event tree analysis. A detailed description of the use of both the "X" and "Y" flags is provided in Appendix A.

4.10.4 Use of Mutually Exclusive Top Events

The Mutually Exclusive Top feature allows the user to define impossible or undesired cutsets and automatically remove them from sequence cutsets. This approach can be more traceable and less tedious than using the cutset editor. As an example to illustrate this feature, consider two pump trains in parallel, and each pump train has a test and maintenance (TM) outage event modeled in the fault tree logic. If the technical specifications did not allow both pumps to be in a TM outage during the operating mode represented in the event tree sequence to be analyzed, then the cutsets produced by the fault tree logic that included the TM of both pumps would not be correct and should be deleted.

To use the Mutually Exclusive Top feature, a fault tree must be built to specify the undesired combinations of events. In our example case, a fault tree named MET could be built that combined TM event 1 and 2 by an AND gate.

Family, MET =
MET AND TMEVENT1 TMEVENT2

Now, from the Link Event Tree menu, select the Generate Sequences option. Choose the sequence or group of sequences for which sequence logic is to be generated, and edit the input screen titled Sequence Generation Information by imputing the fault tree name MET in the field for the Mutually Exclusive Top Name. Now, when IRRAS generates sequence cutsets for the selected sequences, IRRAS will automatically remove the cutsets that have TMEVENT1 and TMEVENT2 occurring together in the same cutset. This was accomplished without having to modify the fault tree logic or having to manually edit sequence cutsets following cutset generation. To review the sequences that have a mutually exclusive top specified, an IRRAS report of the sequence logic can be generated. The mutually exclusive top will appear as a complemented event in the sequence logic report.

4.10.5 Use of Flag Sets

The Flag Set feature automates the ability to specify flag sets that are sequence specific. This feature allows the same fault tree logic to work differently for various situations, depending on the particular setting of the house events in the logic. To illustrate the usefulness of this feature, consider an event tree sequence where the initiating event includes the loss of diesel power, and the fault trees called by the event tree include diesel power dependencies. If the basic event for failure of the diesel to run is DG-FR, then setting the DG-FR calculation type to True (failed) will effectively ensure that the diesel is not credited with successful operation even if there are other basic events that could also cause failure of the diesel generator. Without the Flag Set feature, the user would need to build a change set or modify the database with DG-FR set to True, perform the Generate Changes option, and generate sequence cutsets for only the appropriate sequence or sequences. For each sequence (or group of sequences) having special house event settings, these steps would have to be repeated.

To use the Flag Set feature, the user would build a flag set (it is actually a special change set) in the Generate Event Data menu. To do this, select Add, and enter a name for the flag set. For this example, we will name the flag set FLAGDG. Now, select Probability and modify the house event setting of the selected basic events. In this case, the calculation type for the "current" field under Fault Tree Data for event DG-FR is specified as T for True (meaning guaranteed to fail).

Now, to specify which event tree sequences should use the FLAGDG flag set, go to the Modify Data Base menu, select Event Trees, and choose the Sequence option. In the Edit Sequence menu, use Modify to edit the appropriate sequences by entering the flag set name. The user can review the flag set names specified for each sequence in the Reports menu by generating a sequence logic report. The user can also use the MAR-D extract feature (in the Utilities menu) to extract or load the flag set name using the sequence attribute file (SQA).

Using the Flag Set feature, all sequences in the family can be generated in one step while still ensuring that each sequence uses the proper house event settings. This method is traceable and involves less user manipulation to ensure that cutsets for each sequence are generated with the proper house event settings. Once the sequences have been assigned a Flat Set name, unlike a true Change Set, the Flag Set does not have to be marked to work. It is automatically applied when the sequence Cut Sets are generated.

4.10.6 Use of Importance Measures

Importance measures are sometimes included with the PRA documentation. Importance measures can be used to help determine if sequence cut sets produced by the IRRAS database match the PRA document. IRRAS importance measures can be compared to the PRA document to see if the number of occurrences of each basic event in the PRA sequences is equal to those generated by IRRAS or if there is a mismatch.

5. REFERENCES

1. K. D. Russell and N. L. Skinner, *Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0, Volume 8: Models and Results Database (MAR-D) Reference Manual*, NUREG/CR-6166, EGG-2716, July 1994.

2. K. D. Russell et al., *Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0, Volume 2: Integrated Reliability and Risk analysis System (IRRAS) Reference Manual*, NUREG/CR-6166, EGG-2716, July 1994.

3. R. L. VanHorn, K. D. Russell, and N. L. Skinner, *Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0, Volume 3: Integrated Reliability and Risk analysis System (IRRAS) Tutorial Manual*, NUREG/CR-6166, EGG-2716, July 1994.

4. K. D. Russell et al., *Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0, Volume 1: Technical Reference Manual*, NUREG/CR-6166, EGG-2716, July 1994.

5. Worrell, R. B., *SETS Reference Manual*, NUREG/CR-4213, SAND83-2675, May 1985.

6. T. Ginzburg and W. E. Vesely, *FRANTIC ABC User's Manual*, Applied Biomathematics, Inc., November 1990.

Appendix A

Procedures for Database Loading

Appendix A

Procedures for Database Loading

PROCEDURE 1: ADD AND SELECT A FAMILY

This Procedure Describes Adding and Selecting a Family.

> **IMPORTANT**
>
> The word "Select" as used, for example, in "Select Family" ALWAYS means highlight the option and press <Enter>.

To Add a Family Named SAMPLE from the IRRAS Main Menu Shown in Figure A-1,

1) **Select MODIFY (Modify Data Base).** The Modify Database menu will appear as shown in Figure A-2.

2) **Select Family**. The Edit Family menu will appear as shown in Figure A-3.

3) **Type A** (Add) **and press <Enter>.** The Add Family menu will appear as shown in Figure A-4.

4) **Type the family name** (DATA), **description, and other data in the fields provided.**

5) **Press <Enter>.** Do not type a password.

The IRRAS Main menu will appear as shown in Figure A-1.

To Select the Family Named Sample,

1) **Select SELECT (Select Family).** The Select Family menu will appear as shown in Figure A-5.

2) **Type S, highlight the Sample Family, and press <Enter>.**

The Main menu will appear as shown in Figure A-1, and the selected family name will be displayed on the Main menu.

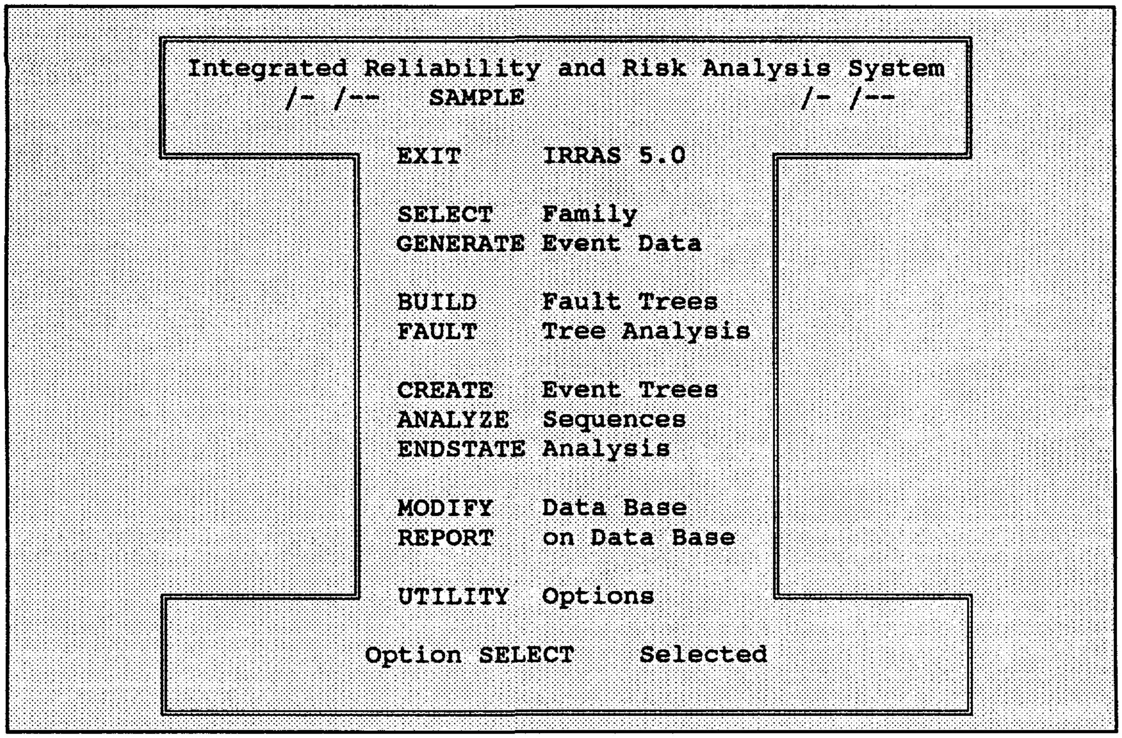


Figure A-1. IRRAS Main menu.

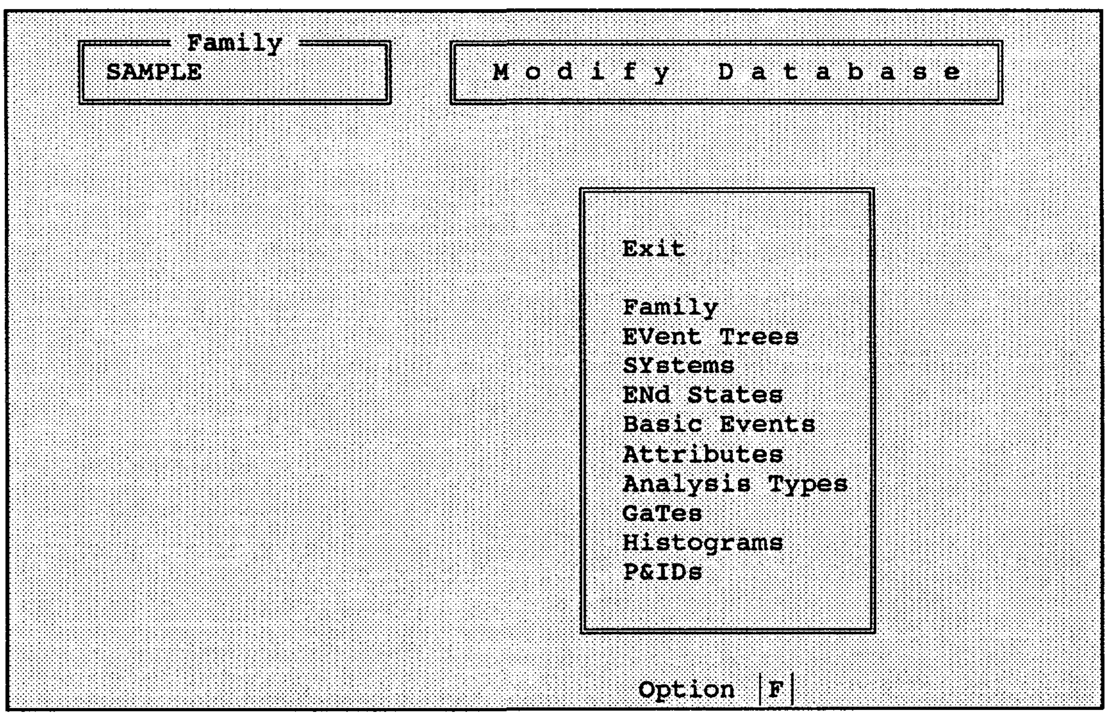


Figure A-2. Modify Database menu.

```
 ═══ Family ═══                 
 SAMPLE                          E d i t   F a m i l y

                  Option |A|  Exit / Add / Modify / Delete / Text

 ═ Family Name ══ Directory ═══════════════ Description ═══════════
 DEMO              DEMO       Demonstration sample family
 LEARN-SARA        LEARN-SA   Sample Family of Data for the Sara 4.0 Tutorial
 SAF50             SAF50

                        <Esc>          <F1>           <F5>
                        Exit           Help           Locate
```

Figure A-3. Edit Family menu.

```
                              A d d   F a m i l y

                        Option |A|  Exit / Add / Passwords

   Name                                 Type                     Tree Type
   Location                             Design
   Company                              Vendor
   Description

       Seismic Histograms
    Low                                 Operational date   ----/--/--
    Medium                              Qualification date ----/--/--
    High                                Mission time       ------E----

                      Press <F1> for Help or List of Values
```

Figure A-4. Add Family menu.

```
 ═══ Family ═══        ┌────────────────────────────┐
 SAMPLE                │  S e l e c t   F a m i l y │
                       └────────────────────────────┘

              Option |S|  Exit / Select / Family Copy / Copy

 ═ Family Name ══ Directory ════════════ Description ═══════════════ St
 DEMO             DEMO        Demonstration sample family              OnL
 LEARN-SARA       LEARN-SA    Sample Family of Data for the Sara 4.0 Tutor OnL
 SAMPLE           SAMPLE      This is sample data base                 OnL

                     <Esc>          <F1>          <F5>
                     Exit           Help          Locate

Key to status: OnL = Online / Cmp = Compressed / CmB = Compressed and Backed Up
               Bck = Backed up without compression
```

Figure A-5. Select Family menu.

PROCEDURE 2: LOADING AND EXTRACTING FLAT FILES

This Procedure Describes Loading and Extracting Flat Files.

Extracting Files

The user may extract flat files from the interactive database by using either MAR-D or IRRAS. All extracted files will be sent to the subdirectory related to the family that is currently selected. The default names for extracted files are shown in Table 2-1, Section 2.4.

To Extract a Flat File,

1) **Select UTILITY (Utility Options)** from the IRRAS Main menu, Figure A-1. The IRRAS Utilities menu will appear as shown in Figure A-6.

2) **Select EXtract (Extract MAR-D Data Files).** The Extract menu will appear as shown in Figure A-7.

3) **Select the MAR-D option from Data to Extract.** The Select MAR-D Data submenu will appear as shown in Figure A-8.

4) **Select the MAR-D data type to extract.** A Select "data type" Data submenu will appear. Figure A-9 is presented as an example of the Select Family Data submenu.

5) **Select the flat file.** IRRAS will prompt the user with messages to help extract the file wanted.

   Caution: IRRAS will overwrite any existing file on the subdirectory with the extracted file of the same name.

```
 Family                 I R R A S   U t i l i t i e s
 SAMPLE

                             Exit

                             Define Constants

                             Load    MAR-D Data Files
                             EXtract MAR-D Data Files

                             Archive Data Base

                             Version 1.0/5.0 Interface

                             Recover Data Base

                                  Option |E|
```

Figure A-6. IRRAS Utilities menu.

```
 Family                  E x t r a c t
 SAMPLE

 DATA to EXTRACT

    Exit

    MAR-D
      IRRAS
      SARA
    SETS

   Option |E|
```

Figure A-7. Extract menu.

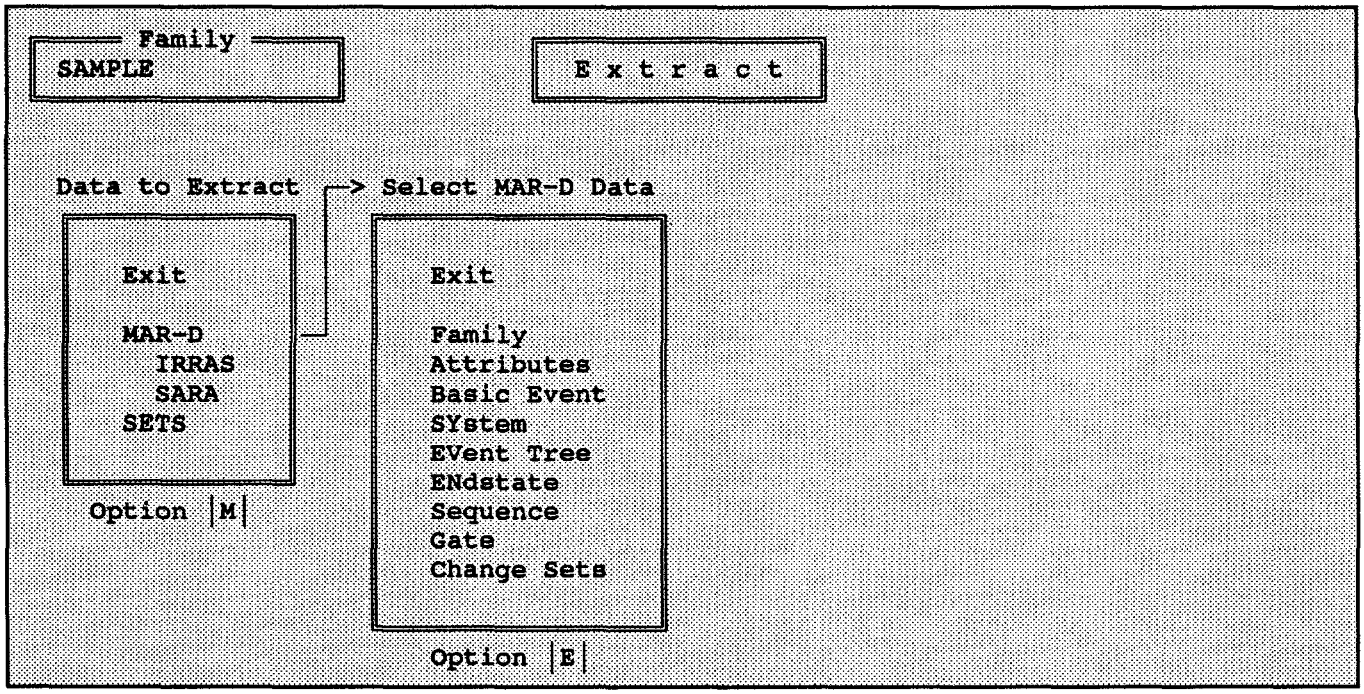


Figure A-8. Select MAR-D Data submenu.

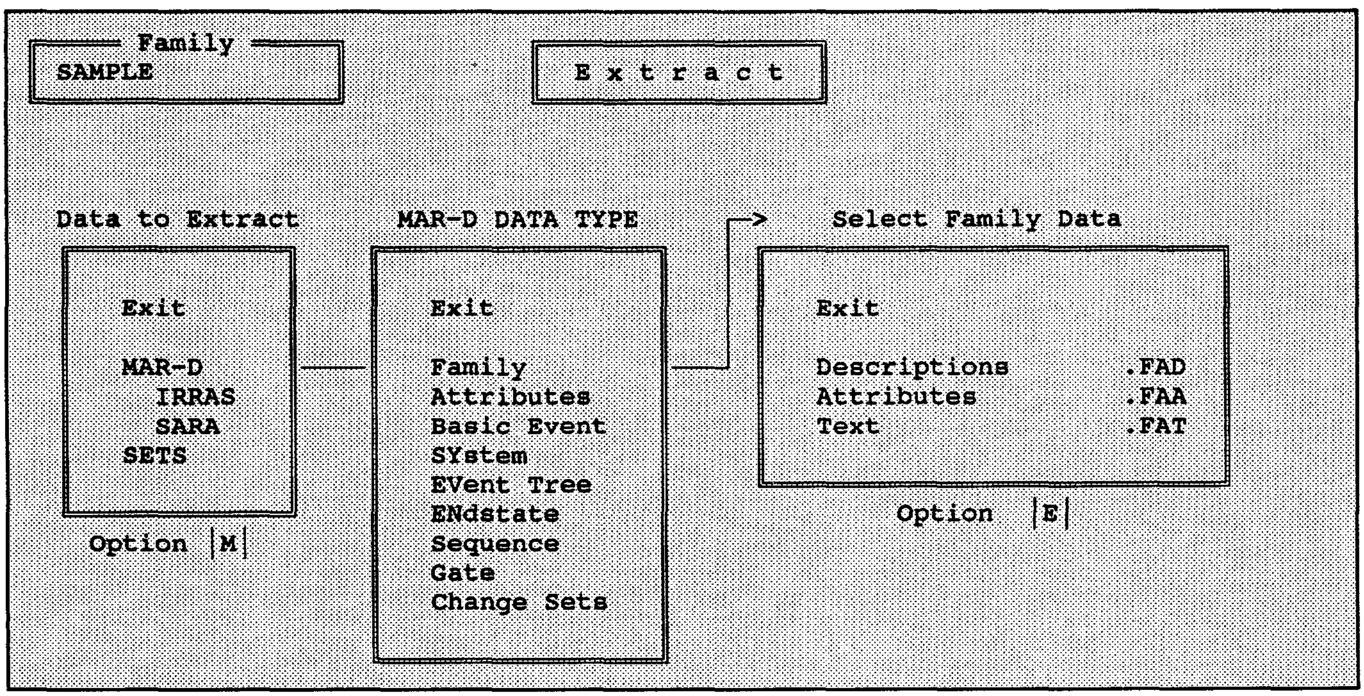


Figure A-9. Select Family Data submenu.

Loading Files

The user may also use the Modify Event Data module to enter data into the interactive database. After creating flat files in an ASCII format, these files can be loaded into the database from the subdirectory related to the selected family.

To Load a Flat File,

1) **Select UTILITY (Utility Options)** from the IRRAS Main menu, Figure A-1. The IRRAS Utilities menu will appear as shown in Figure A-6.

2) **Select Load (Load MAR-D Data Files).** The Load menu will appear as shown in Figure A-10.

3) **Select the MAR-D option from Data Source.** The Select MAR-D Data submenu will appear as shown in Figure A-11.

4) **Select the MAR-D data type to load.** A Select "data type" Data submenu will appear. Figure A-12 is presented as an example of the Select Family Data submenu.

5) **Select the flat file.** IRRAS will prompt the user with messages to help load the file wanted.

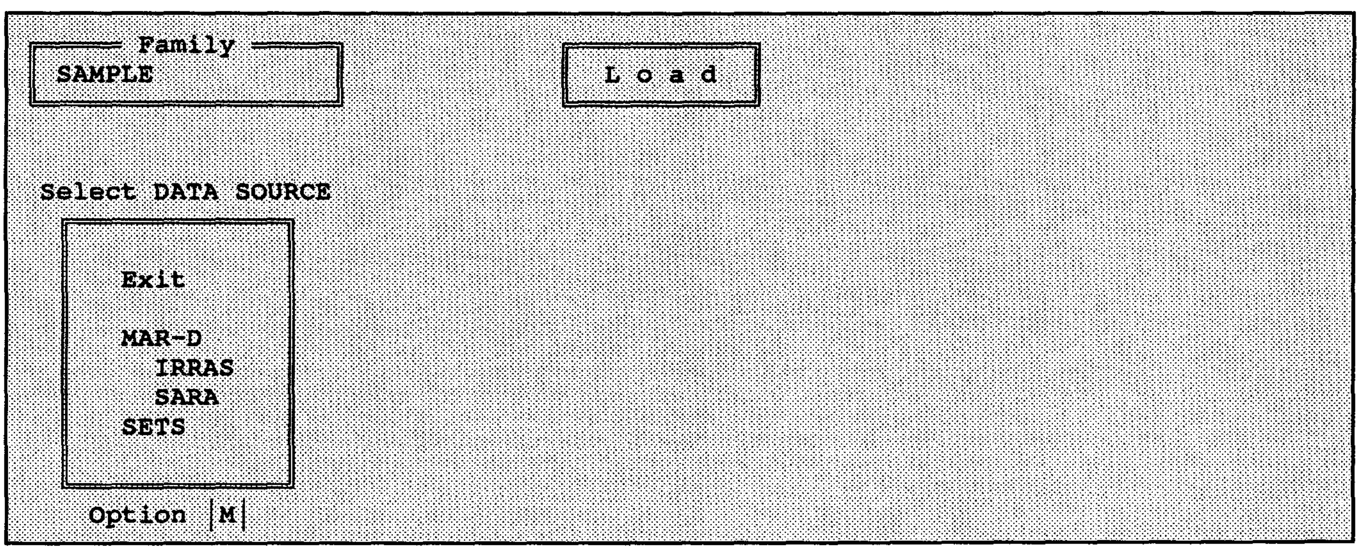


Figure A-10. Load menu.

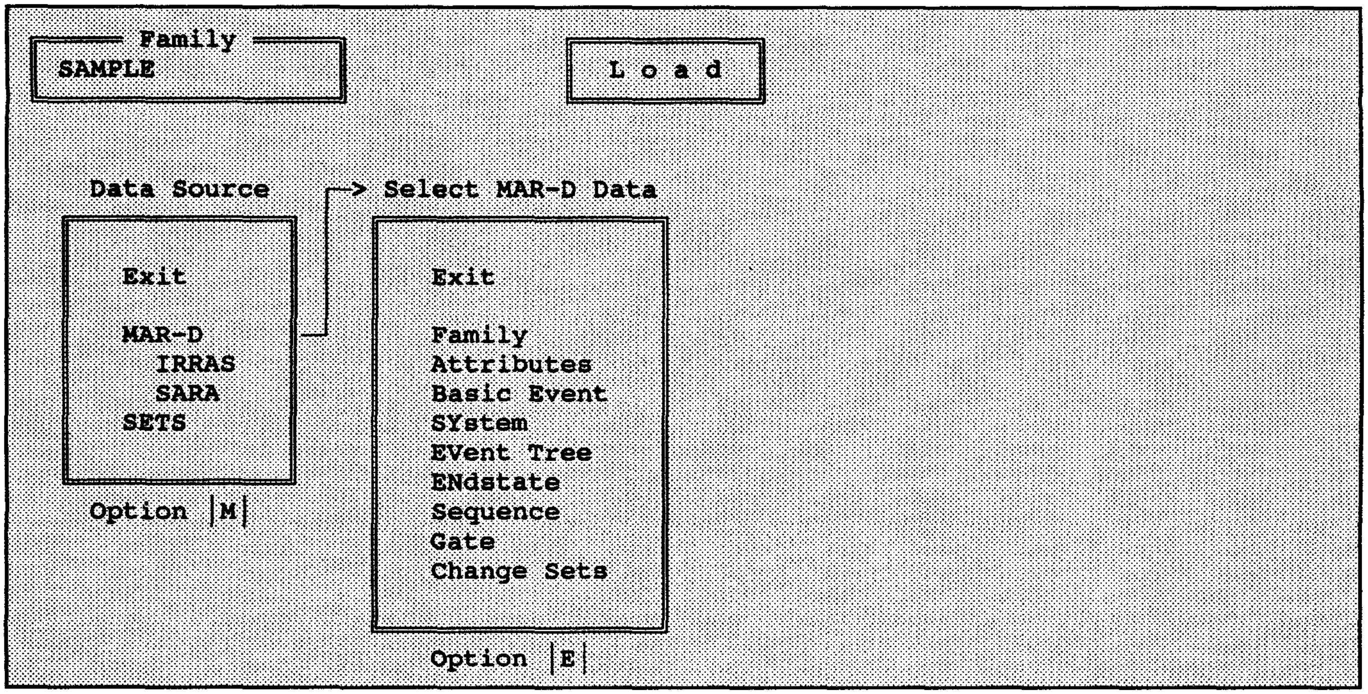


Figure A-11. Select MAR-D Data submenu.

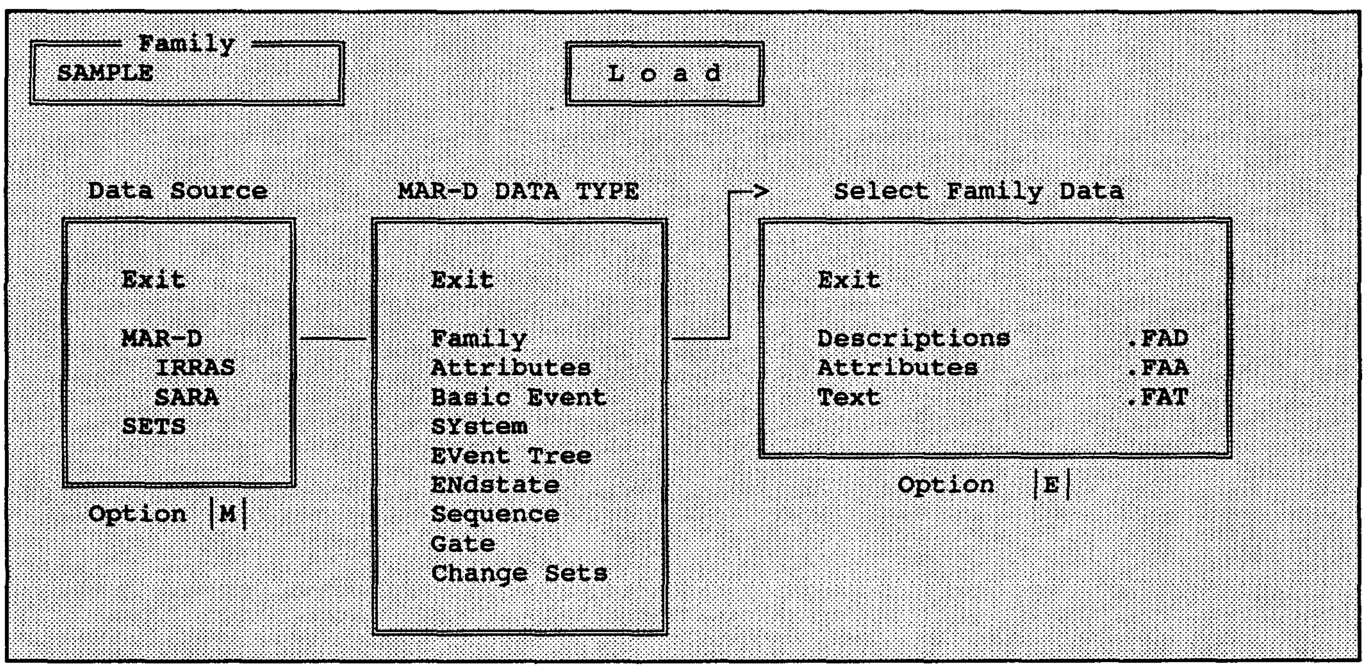


Figure A-12. Select Family Data submenu.

PROCEDURE 3: ALPHA TO GRAPHICS CONVERSION PROCESS

This Procedure Describes the Alpha to Graphics Conversion.

The fault tree data entry is complicated by the fact that IRRAS utilizes an interactive database. Information entered in the process of graphical fault tree construction is implemented in many areas of the program. Graphical data structure is translated into logic, and other information is entered into the interactive database using internal lists. Such information includes the type of gates and basic events used, the textual descriptions entered in gate and basic event boxes, and the textual descriptions added for a fault tree description. The information on these internal lists can subsequently be extracted into IRRAS flat files. Conversely, IRRAS can be used to build fault tree graphics from logic and descriptions entered in the database using the alpha to graphics conversion.

It is important to note that for cut set generation and quantification, IRRAS uses only the logic and not the graphical representation of the fault tree. IRRAS can create logic from any fault tree graphics that are built, and the graphics are useful for easy visualization of the system.

When a newly developed fault tree is graphically saved, a .DLS file is automatically created on the family subdirectory. The graphics file is translated into internal fault tree logic. Fault tree descriptions, basic event names and descriptions, and gate names, descriptions, and attributes are appended to IRRAS internal listings.

When a fault tree is built and saved initially, IRRAS will place the top-most textual description (16 characters or longer) from the fault tree into the system description. If no text is given but a description is given for the top gate, this will be placed into the system description. If no text is given and there is no description for the top gate, the system description will be left blank.

The system description is directly associated with the graphics. If an alpha to graphics conversion is done, IRRAS will use the system description to create or update a title in the fault tree graphic.

The .DLS file contains fault tree graphical information. To view and modify a fault tree, the file (.DLS) for that fault tree must be available on the subdirectory. Once the graphics logic has been loaded into the interactive database, it is not necessary to have the graphics available (for cut set generation and quantification). The .DLS files can be cleared and extracted utilizing the Extract Graphic Fault Trees option from the Build option.

To Use the Alpha to Graphics Conversion,

1) Load the fault tree, gate, and basic event files into the database, including:

  a) .FTD (fault tree descriptions)

  b) .FTL (fault tree logic)

c) .GTD (gate descriptions)

d) .BED (basic event descriptions)

2) Perform the alpha to graphics conversion:

a) Make sure the IRRAS Main menu is displayed.

b) Select BUILD **(Build Fault Trees)** (see Figure A-13). The IRRAS fault tree graphics system menu will appear.

c) Select the **Alpha to Graphics** option. The alpha to graphics menu will appear. (See Figure A-14)

d) Type **<C>** to select the Convert option. Highlight the fault tree to be converted.

e) Press **<Enter>**. IRRAS will prompt the user concerning the use of tables and boxed events. Press **<Enter>** to continue.

f) Continue converting files as necessary.

```
 Family
 DEMO

                         I R R A S
                Fault Tree Graphics System

          Exit

          Build Graphic Trees

          Plot Trees
          Load Graphic Trees
          Alpha to Graphics
          Graphical Pager

          eXtract Fault Trees
          Define Plotter Pens

                 Option |E|
```

Figure A-13. Build Fault Trees submenu.

```
 Family
 DEMO                 A l p h a   t o   G r a p h i c s

               Option |C| Exit / Convert

 N a m e                  D e s c r i p t i o n
   CCS              Containment Cooling System fault tree
   ECS              Emergency Cooling System fault tree

   <Esc>      <F1>      <F2>        <F3>       <F4>        <F5>
   Exit       Help    Mark File   Mark All   Mark Range   Locate
```

Figure A-14. Alpha to Graphics submenu.

PROCEDURE 4: GENERATE EVENT DATA

This Procedure Describes the Process of Generating Event Data. This Process will Update the Current Case Values with the Base Case Values.

All basic event data entered in the IRRAS Modify Event database is automatically placed in the base case database. When loading values, IRRAS will allow information update to the base case and/or the current case database. Unless queried during the process, any analysis performed using the IRRAS program defaults to values and/or cut sets drawn from the current case database. Note that basic fault tree and event tree logic remains the same for both cases.

To Select the Generate Event Data Module from the Main Menu,

1) **Select GENERATE (Generate Event Data).** The Change Sets menu will appear as shown in Figure A-15.

2) **Type G,** (Generate Changes) **and press <ENTER>.** ("Enter Mission Time for Generation" and "Propagate Event Failure (Y/n) N" will appear in a box at the bottom of the screen.)

3) **Press <ENTER>.** ("Event data set to base case (no changes)" will appear at the bottom of the screen.)

4) **Type E (Exit) and press <ENTER> to exit to the Main menu.** (The Esc key can also be used to exit to the Main menu.)

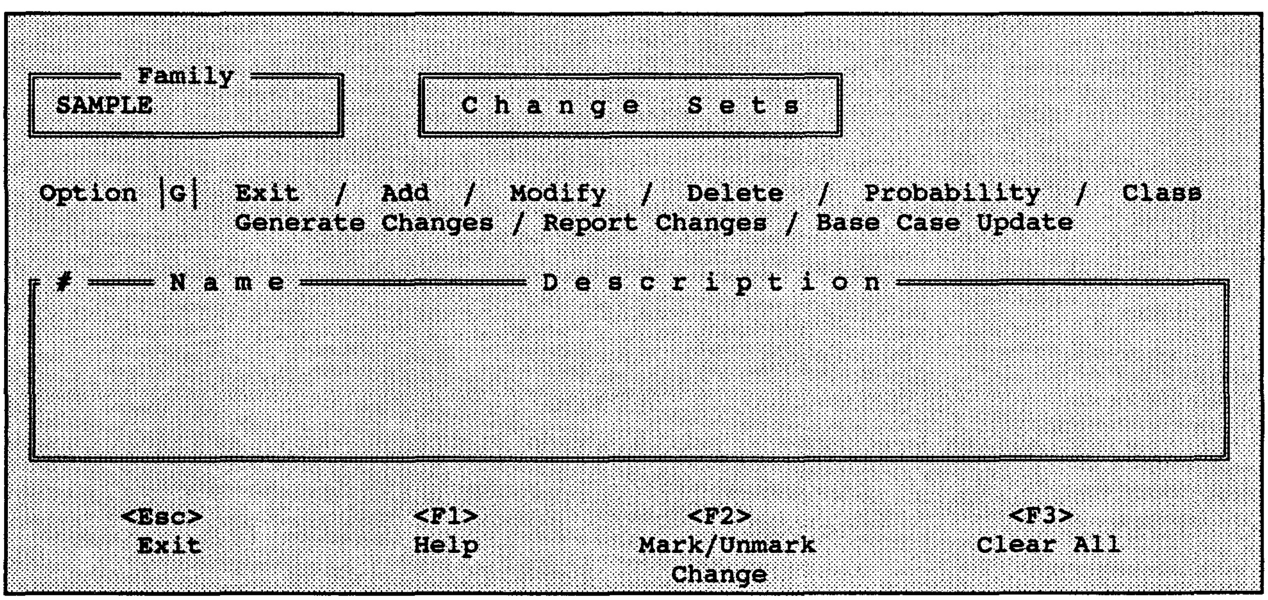


Figure A-15. Change Sets menu.

PROCEDURE 5: CHANGE SETS

This Procedure Describes the Process of Adding and Marking a Change Set Called CS-1.

Implementing a change set allows the user to perform a sensitivity study. The event values in the base case data are modified during the generate changes processes for use in the current case data.

This process includes:

Adding the change set
Making a class change to the CS-1 change set
Making a probability change to the CS-1 change set
Marking the CS-1 change set and generating changes.

Change sets are used to manipulate the base case event data to examine the changes in the probabilities of plant accidents and accident sequence failures based on basic events. The user may also create change sets to be applied to basic events for later propagation through sequence cut sets generation.

IMPORTANT

IRRAS contains information in two databases, the current (or working case) and the base case. They are not necessarily the same. The user may update and change the base case data to the current case for analysis without modifying the base case data with a change set.

Current case data are ALWAYS used in any cut set quantifications.

Adding a Change Set

To Add a Change Set Named CS-1,

1) **Select GENERATE (Generate Event Data).** The Change Sets menu should appear as shown in Figure A-15.

2) **Type A** (Add) **and press <Enter>.** The Add Change Set menu will appear.

3) **Type the change set name** (CS-1) **and a description of your choice in the fields provided.**

4) **Press <Enter>.** ("*New change record added* " will appear on the bottom of the screen, and the change record name will appear under the Name section of the Change Sets menu.) The Change Sets menu will appear.

Making a Class Change

This option allows the user to change event data parameters for a specified grouping of events. The event class is defined by entering data in the Event Attributes data fields. The more of these fields that are filled in, the finer the class definition becomes.

To use this option, the user must have already added a change set as described above.

To Make a Class Change,

1) **Type C** (Class), **highlight CS-1 with the cursor, and press <Enter>.** The Class Change menu will appear as shown in Figure A-16.

2) **Type NAME* as the primary <P> name (under Names) in the Event Attributes Mask section of the Class Change menu.** (The asterisk (*) is a wildcard that acts as a substitute representing a whole word or a group of characters.)

3) **Move the cursor down to the Probability blank in the Failure Data section and type the new Probability).**

4) **Press <Enter>.** ("*Class change added* " will appear on the bottom of the screen.) The Change Sets menu will appear.

```
┌──── Family ────┐   ┌───────────────────────┐   ┌──── Change Set ───┐
│ SAMPLE         │   │ C l a s s   C h a n g e │   │ SAMPLE            │
└────────────────┘   └───────────────────────┘   └───────────────────┘

┌─────────────────────── Event Attributes Mask ────────────────────────┐
│      Names                                      Suscept.             │
│  <P>                    Comp Id   System   Train   1  2  3  4  5  6  7  8  │
│  <G>                                                N  N  N  N  N  N  N  N  │
│ Uncert.Corr.  Category     Type   F/Mode  Location  9 10 11 12 13 14 15 16 │
│ Class                                               N  N  N  N  N  N  N  N  │
└─────┬──────── Failure Data ─────────┬────── Miscellaneous ──────┬────┘
      │ Calc. Type                    │ Process Flag              │
      │ Probability ------E----       │                           │
      │ Lambda      ------E----       │ Note: Leave values blank if │
      │ Tau         ------E----       │       no changes are desired. │
      │ Mission Time ------E----      │                           │
      ├──────── Uncertainty Data ─────┼──── Seismic Fragility ────┤
      │ Dist. Type                    │ Calc. Type                │
      │ Name                          │ Name                      │
      │ Value 1     ------E----       │ Beta Random  ------E----  │
      │ Value 2     ------E----       │ Beta Uncert. ------E----  │
      │ Corr. Class                   │ Acceleration ------E----  │
      │                               │ Corr. Class               │
      └─────────────── Press <F1> for Help ───────────────────────┘
```

Figure A-16. Class Change menu.

Making a Probability Change

This option gives the user the flexibility to experiment with setting different basic event failure and uncertainty data. These data values may be set for a single event or for a specified group of events. The option also provides a reset option to set data values back to the base case values.

To use this option, the user must have already added a change set as described above.

To Make a Probability Change,

1) **Type P** (Probability), **highlight CS-1 with the cursor, and press <Enter>.** The Select Event menu will appear.

2) **Type P** (Probability), **highlight a basic event with the cursor, and press <Enter>.** The Event Probability Changes menu will appear.

3) **Move the cursor down to the Current Probability blank in the Failure Data section and type the new Probability.**

4) **Press <Enter>.** (A large "P" will appear in the left-hand column by the basic event. Notice that a large C is present in the left-hand columns when a class change has been used. Note the symbol explanation at the bottom of this screen.)

5) **Type E (Exit) and press <Enter> to exit to the Change Sets menu (or press the Esc key).**

Or

If the user wants to continue the probability change process, **repeat steps #3 and #4 immediately after step #4** since the user is still in the Probability Change mode.

Marking a Change Set

Using the <F2> key allows the user to mark/unmark the change sets to be used during the generation process. If more than one change set is marked, then the probability and class changes in the change sets marked with the highest number will take precedence over any change from lower numbered change sets.

To use this option, the user must have already added a change set as described above. Additionally, either a class or probability change should be implemented.

To Mark a Change Set (CS-1) from the Change Set Menu,

1) **Highlight CS-1 with the cursor and press <F2>.** A number (1) should appear in the far-left corner by the change set.

2) **Type G**, (Generate Changes) **and press <Enter>.** ("Enter Mission Time for Generation" and "Propagate Event Failure (Y/n) N" will appear in a box at the bottom of the screen.)

3) **Press <Enter>.** ("Event data set to base case (no changes)" will appear at the bottom of the screen.)

4) **Type E (Exit) and press <Enter> to exit to the main menu (or press the Esc key).**

IMPORTANT

The change set will remain marked until the user unmarks it by using the <F2> key (mark/unmark) or the <F3> key (clear all).

Marked sets are ALWAYS used to generate changes.

PROCEDURE 6: X AND Y FLAGS

This Procedure Describes the Process of Using the X and Y Process Flags.

Using the "Y" Process Flag

This section discusses the concept and use of the "Y" process flag. The Process Flag options appear in the Modify Event menu (Figure A-17).

1) Add and select a family called COINTOSS.

2) Once in this database, the user will need to create the event tree shown in Figure A-18. This event tree will calculate the probability of combinations of heads (H) and tails (T) from tossing a coin twice. Possible combinations that can occur are HH, HT, TH, or TT. This can be observed in the sequence and endstate names shown in Figure A-18.

Since this is a simple problem, the probability for all possible endstates can be hand calculated. As is shown in Table A-1, the probability of tossing two heads is 0.25, the probability of tossing a head and a tail is 0.5 and the probability of tossing two tails is 0.25. This sums to 1.0 across all the possible sequences.

```
 Family
 COINTOSS                    M o d i f y   E v e n t

                         Option |M|   Exit / Modify

                              Event Data
 Names Primary TOSS1                 Process      Category      Component
     Alternate TOSS1                 Flag                       Id
         Group
   Description
       Failure Data            Seismic Fragility             Attributes
 Calculation Type      1  Type   Name                       System
 Probability   5.000E-001 Random Beta      ------E----      Train
 Lambda       +0.000E+000 Uncertainty Beta ------E----      Type
 Tau          +0.000E+000 F. Acceleration  ------E----      Failure Mode
 Mission Time +0.000E+000 Correlation Class                 Location
     Uncertainty Data                 Transformations
 Distribution Type     L          Type                Level  ---
 Name     Log Normal                  Susceptibilities
 Value 1       1.000E+000  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 Value 2       ------E---- N  N  N  N  N  N  N  N  N  N  N  N  N  N  N  N
 Correlation Class
                  Press <F1> for Help or List of Values
```

Figure A-17. IRRAS 5.0 Modify Event Menu.

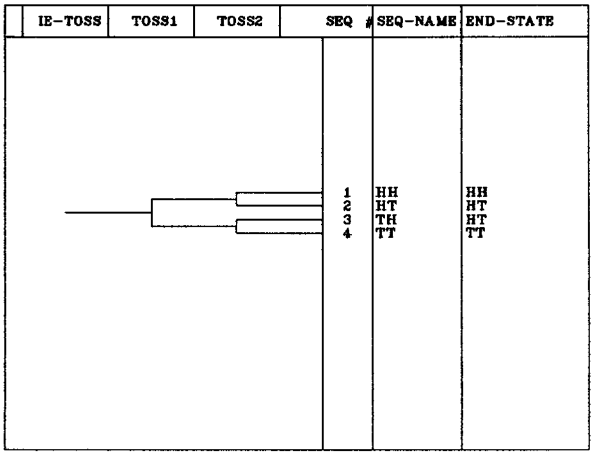


Figure A-18. Tosstree event tree.

Table A-1. IRRAS sequence cut sets report using default.

```
                    SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                 Family: COINTOSS             Event Tree: TOSSTREE
                   Sequence: 1                Init. Event: IE-TOSS
                          Mincut Upper Bound  1.000E+000

Cut    %   % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  1.0E+000 <PASS>

                    SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                 Family: COINTOSS             Event Tree: TOSSTREE
                   Sequence: 2                Init. Event: IE-TOSS
                          Mincut Upper Bound  5.000E-001

Cut    %   % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  5.0E-001 TOSS2

                    SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                 Family: COINTOSS             Event Tree: TOSSTREE
                   Sequence: 3                Init. Event: IE-TOSS
                          Mincut Upper Bound  5.000E-001

Cut    %   % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  5.0E-001 TOSS1

                    SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                 Family: COINTOSS             Event Tree: TOSSTREE
                   Sequence: 4                Init. Event: IE-TOSS
                          Mincut Upper Bound  2.500E-001

Cut    %   % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  2.5E-001 TOSS1, TOSS2
```

3) Enter the top event values. In the process of creating this event tree, IRRAS has placed IE-TOSS as an initiating event and two top events (TOSS1 and TOSS2) as developed events into the Basic Event listing (under Modify Database). For calculation, it is necessary to enter the probability of a failure for TOSS1 and TOSS 2 as 0.5. Since IRRAS looks at failure space, this will be the probability of a tail given an unbiased coin. Leave the process flag space blank. The initiating event IE-TOSS should be 1.0 as default. Accept the default.

4) Generate changes (by selecting G in the Generate Event Data module) to update the probability values that have just been entered into the base case to the current (or working) case.

5) Generate sequences (by selecting G in the Link Event Trees module under Analyze Sequences). Type "CON" in report name on the Sequence Generation Information. This will cause the sequences generated to appear on the screen.

6) Generate sequence cutsets (by selecting G in the Analyze Sequence). Accept the IRRAS defaults.

7) View the results. As shown in Table A-1, the results are not what was anticipated. This is because, by default, IRRAS will generate sequence cutsets utilizing both a success and failure logic. This will include either the developed event or fault tree (if one was associated with that top event). But, then only the failure logic is presented and the failure probabilities are used in the quantification process.

   Entering a process flag will allow the user to indicate exactly how IRRAS is to handle developed events (and/or fault trees). The "Y" process flag uses the developed event value for the probability of failure and 1 minus the probability of failure of the developed event for the success.

8) Enter a "Y" as the process flag for both TOSS1 and TOSS2. Repeat steps 4 to 7 and view results. As shown in Table A-2, the calculated results are now correct.

Table A-2. IRRAS sequence cut sets report using "Y" process flag.

```
                    SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                  Family: COINTOSS              Event Tree: TOSSTREE
                    Sequence: 1                 Init. Event: IE-TOSS
                           Mincut Upper Bound  2.500E-001

Cut    %    % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  2.5E-001 /TOSS1, /TOSS2

                     SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                  Family: COINTOSS               Event Tree: TOSSTREE
                    Sequence: 2                 Init. Event: IE-TOSS
                           Mincut Upper Bound  2.500E-001

Cut    %    % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  2.5E-001 /TOSS1, TOSS2

                     SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                  Family: COINTOSS              Event Tree: TOSSTREE
                    Sequence: 3                 Init. Event: IE-TOSS
                           Mincut Upper Bound  2.500E-001

Cut    %    % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  2.5E-001 TOSS1, /TOSS2

                     SEQUENCE CUT SETS (QUANTIFICATION) REPORT
                  Family: COINTOSS              Event Tree: TOSSTREE
                    Sequence: 4                 Init. Event: IE-TOSS
                           Mincut Upper Bound  2.500E-001

Cut    %    % Cut   Prob/
No.  Total  Set    Freq.                ALTERNATE CUT SETS
---- ----- ----- --------- -----------------------------------------------
   1 100.0 100.0  2.5E-001 TOSS1, TOSS2
```

Using the "X" Process Flag

The "X" flag behaves in the following manner. With the process flag for an event tree top event set to "X", IRRAS links in the associated system success and failure fault trees and solves the boolean logic. When reporting the results, successful system basic events are ignored although they have been accounted for in the logic.

PROCEDURE 7: RECOVERY RULES

This Procedure Describes the Process of Developing and Using the Recovery Rules.

This section discusses the concept, development, and use of the recovery rules. The Recovery Rules options appear in the Fault Tree Analysis menu (Figure A-19) for fault tree rules and the Analyze Sequence menu (Figure A-20) for event tree rules. Recovery rules can be applied for (1) all systems (i.e., family system rules), (2) a single system (i.e., fault tree rules), (3) all sequences (i.e., family sequence rules), (4) a single event tree (i.e., event tree rules), or (5) a single sequence (i.e., sequence rule). The discussion of these rules will be by way of a few simple examples of the rules along with a detailed overview of the recovery rules (see Table A-3).

The SAPHIRE 5.0 recovery rules are "free-form" logic rules that allow the user to alter or delete system or sequence cutsets. Although called "recovery rules," the recovery rules have developed from the simple addition of recovery events onto specified cutsets into a powerful rule-based system for cutset manipulation. Thus, the "recovery rules" can now be used for advanced probabilistic risk assessment techniques such as (1) the automated addition of sequence recovery events, (2) the addition of common-cause cutsets, and (3) the elimination of mutually-exclusive events (e.g. restricted or impossible combinations of events).

The rules are entered in a free-form text editor within IRRAS (similar to KEDIT or the DOS text editor). The rules have a very specific format and have certain command keywords that can be used. The structure of the rules and the keywords that are available are discussed below. It should be pointed out that the rules can be exported and loaded through MAR-D. Procedures for exporting and loading files are discussed in the MAR-D Reference Manual.[1]

The rules follow a format similar to the structure that is found in traditional programming languages (e.g., BASIC, MODULA-2, or C). As such, the ability exists to define "macros" and "if...then" type structures. But, before discussing the particular structure of the rules, it is best if the keywords and symbols were defined. Table A-3 contains a list of keywords and symbols that are used in the rule editor. This table also includes the definition and usage of each keyword and symbol. Within the "usage" column in Table A-3, the particular keyword or symbol that is being presented is shown in bold face. Words or phrases that are italicized are intended to represent a particular command or group of commands and, as such, should not be included as part of the rule. Instead, an appropriate command (e.g., a specified search criteria, a keyword, or a logic expression) should replace the italicized text.

Now that the keywords and symbols have been defined, the structure of the rules will be discussed. This discussion will take place by way of specific examples.

The first example demonstrates how the recovery rules can be used to add recovery actions. The rule for Example 1 adds the recovery action NRAC-12HR to every cutset for a particular sequence. Consequently, this rule would have to be typed into the event tree *sequence* rule editor for the sequence of interest. Also, from within the rule editor, putting the cursor on NRAC-12HR and pressing <ALT>R will add the event into the database as a recovery action.

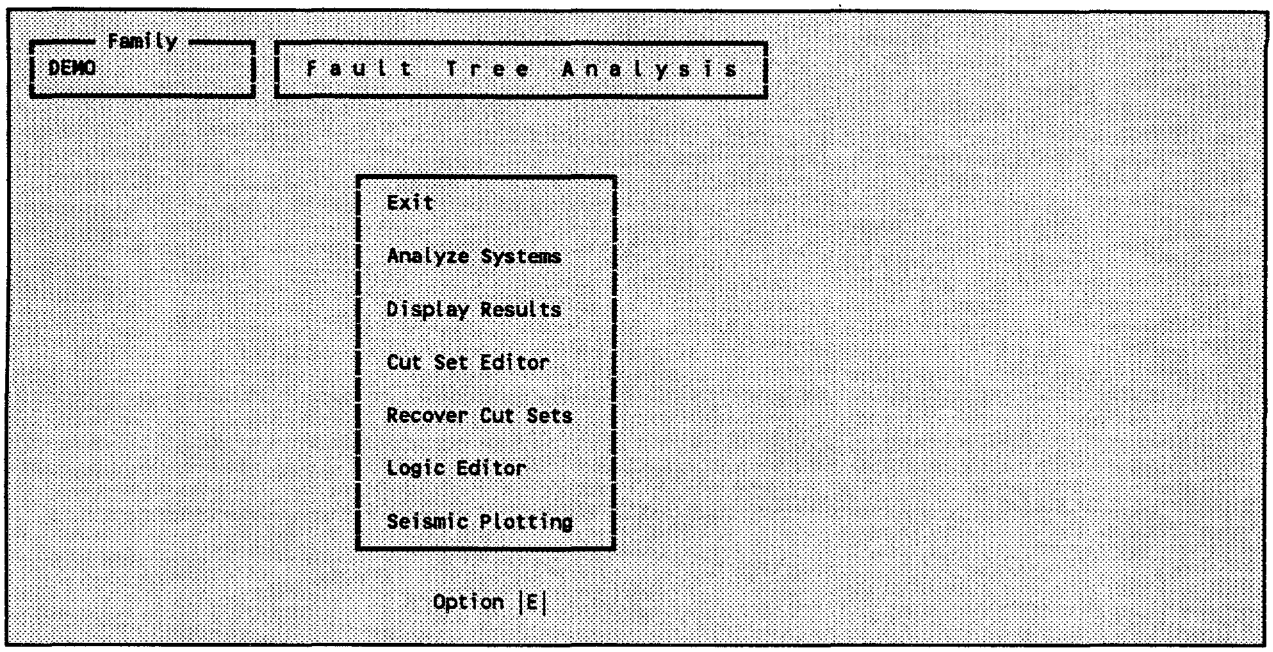


Figure A-19. Fault Tree Analysis menu.

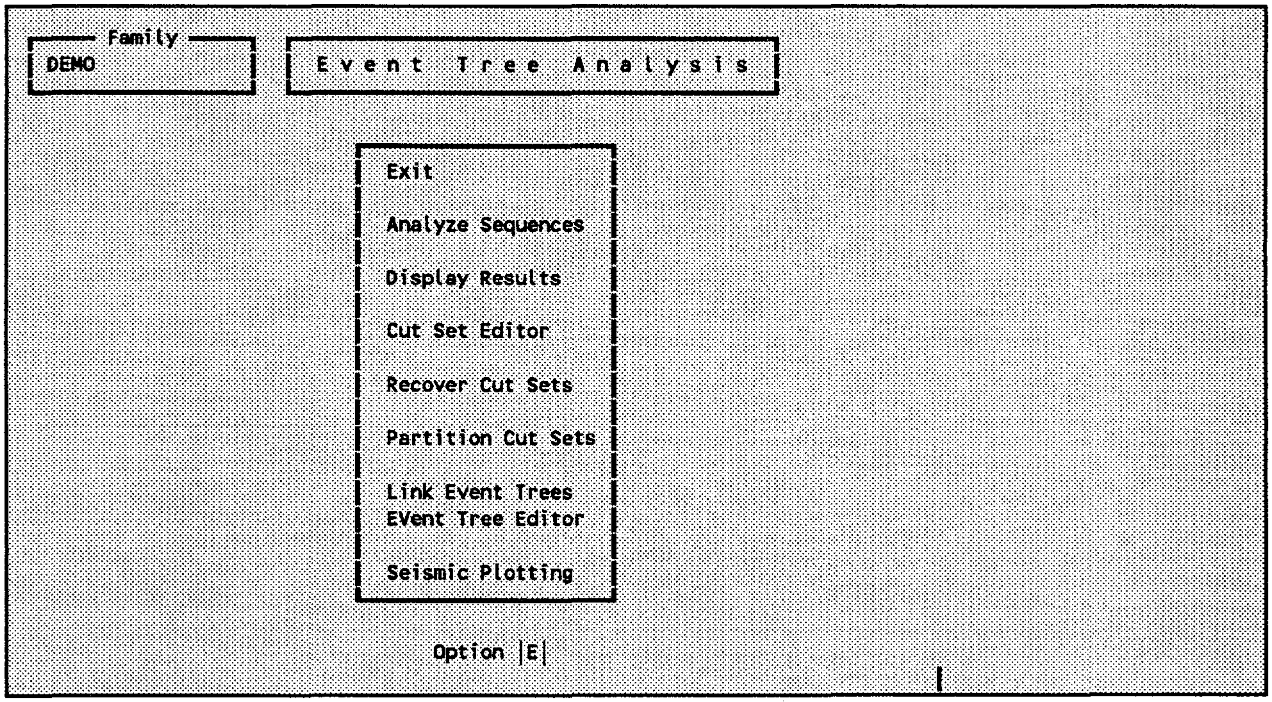


Figure A-20. Analyze Sequences menu.

Table A-3. List of keywords or symbols used in the SAPHIRE 5.0 recovery rules.

Keyword or symbol	Definition	Usage

if then	Keyword pair that indicates that a search criteria is being specified. Note that the keywords are lower case.	**if** *"search criteria"* **then**
		perform some action on each cutset...;
		endif

endif	Keyword that indicates the end of a particular rule. Note that the keyword is one word and is lower case.	if *"search criteria"* then
		perform some action on each cutset...;
		endif

always	Keyword that indicates that every cutset that is being evaluated satisfies the search criteria. Note that the keyword is lower case.	if **always** then
		perform some action on each cutset...;
		endif

recovery =	Keyword that indicates that a recovery event is going to be added to the cutset being evaluated. Note that the keyword is lower case.	if *"search criteria"* then
		recovery = NAME-OF-RECOVERY-EVENT;
		endif

init( )	Keyword used in the search criteria to indicate that a sequence cutset has a particular initiating event. Note that the keyword is lower case.	if **init**(INITIATOR-NAME) *"and other search criteria"* then
		perform some action on each cutset...;
		endif

~	Symbol used in the search criteria to indicate that a particular event *will not* be in the cutset that is being evaluated.	if (~SEARCH-CRITERIA) *"and other search criteria"* then ... The search criteria will be satisfied for all cutsets that do not contain SEARCH-CRITERIA (and also contains the optional "other search criteria"). SEARCH-CRITERIA may be either an initiating event, basic event, macro, or logic expression.
/	Symbol used to represent a complemented event (i.e., the success of a failure basic event).	if (/BASIC-EVENT) *"and other search criteria"* then The search criteria will be satisfied for all cutsets that contain the complement of BASIC-EVENT (and also contains the optional "other search criteria").
|	Symbol used to represent a comment contained in the rules. Everything on a line to the right of this symbol will be ignored by the rule compiler.	| Place your comments here! | Note that blank lines are also permissible!
;	Symbol to indicate the end of a macro line or a line that modifies the cutset being evaluated.	| *usage for a macro command* MACRO-NAME = *"search criteria"* **;** | *usage for a cutset modification line* recovery = RECOVERY-EVENT **;**

Table A-3. (continued).

Keyword or symbol	Definition	Usage
*	Symbol to indicate the logical AND command.	if SEARCH-CRITERIA1 * SEARCH-CRITERIA2 then The search criteria will be satisfied for all cutsets that match SEARCH-CRITERIA1 and SEARCH-CRITERIA2. The SEARCH-CRITERIA# may be either an initiating event, basic event, macro, or logic expression.
+	Symbol to indicate the logical OR command.	if SEARCH-CRITERIA1 + SEARCH-CRITERIA2 then The search criteria will be satisfied for all cutsets that match either SEARCH-CRITERIA1 or SEARCH-CRITERIA2. The SEARCH-CRITERIA# may be either an initiating event, basic event, macro, or logic expression.
( )	Symbols to indicate a specific grouping of items.	if (A + B) * (C + D) then The search criteria above would return all cutsets that contain: [A * C], [A * D], [B * C], or [B * D].

AddEvent =	Key word that indicates that an event will be added to the cutset being evaluated. Note the particular capitalization of the keyword.	if *"search criteria"* then
		AddEvent = EVENT-NAME;
		endif

DeleteEvent =	Keyword that indicates that an event will be deleted from the cutset being evaluated. Note the particular capitalization of the keyword.	if *"search criteria"* then
		DeleteEvent = EVENT-NAME;
		endif

NewCutset;	Keyword that indicates that a new, empty cutset will be added to the list of cutsets. This new cutset will then become the cutset that is being evaluated. Note the particular capitalization of the keyword.	if *"search criteria"* then
		NewCutset;
		now make additions to the empty cutset...
		endif

DeleteRoot;	Keyword that indicates that the original cutset (i.e., the cutset that satisfied the search criteria) will be deleted. Note the particular capitalization of the keyword.	if *"search criteria"* then
		DeleteRoot;
		endif

CopyCutset;	Keyword that indicates that the cutset being evaluated will be copied and added to the list of cutsets. This copied cutset will then become the cutset that is being evaluated. Note the particular capitalization of the keyword.	if *"search criteria"* then
		CopyCutset;
		now make modification to a copy of the cutset...
		endif

Table A-3. (continued).

Keyword or symbol	Definition	Usage

CopyRoot;	Keyword that indicates that the original cutset (i.e., the cutset that satisfied the search criteria) will be copied. This copied cutset will then become the cutset that is being evaluated. Note the particular capitalization of the keyword.	if *"search criteria"* then
		CopyRoot;
		now make modifications to a copy of the original cutset...
		endif

MACROS	A macro is a user-definable keyword that specifies specified search criteria. Note that the macro name must be all upper-case, must be 16 characters or less, and must not include any of the SAPHIRE-restricted characters (e.g., a space, *, ?, \, /). The macro definition line can wrap around to more than one line, but the end of the macro must be indicated with a semicolon.	**MACRO-NAME** = SEARCH-CRITERIA;
		
		if **MACRO-NAME** *"and other search criteria"* then
		perform some action on each cutset...;
		endif
		
		The SEARCH-CRITERIA may be either an initiating event, basic event, macro, or logic expression.

Example 1

```
| A rule to apply NRAC-12HR recovery event to all
| cutsets in the sequence.
if always then
  recovery = NRAC-12HR;
endif
```

The second example demonstrates how recovery actions can be added to certain cutsets in a particular sequence. This example will be added as an event tree family rule (meaning it may affect all sequences in the family), but will key on only one of two different initiating events. Also, this example will demonstrate the use of a macro (the macro is called KEY-ON-INIT).

Example 2

```
| Define a macro in order to pick up only those
| sequences that have LOSP or SBO as initiators.
KEY-ON-INIT = init(LOSP) + init(SBO);
| Search on either the LOSP or SBO and basic events.
| DG-A or DG-B.
if KEY-ON-INIT * (DG-A + DG-B) then
  recovery = NRAC-12HR;
endif
```

The third example demonstrates how the "recovery" rules could be used to add common-cause events to the cutsets. Example 3 defines a search criterion that identifies the failure combination of two auxiliary feedwater pumps (pump A and pump B). If these two basic events are found in a cutset, then a new cutset will be created that replaces the independent failures of the two pumps with a single common-cause basic event. This rule could be placed in either (or both) the system family rules or the event tree family rules.

Example 3

```
| Define a macro in order to pick up only those cutsets that
| have combinations of AFW-PUMP-A and AFW-PUMP-B.
CCF-AFW-PUMPS = AFW-PUMP-A * AFW-PUMP-B;
| Search for the AFW pump basic events and replace
| with a CCF event.
if CCF-AFW-PUMPS then
  | First make a copy of the original cutset.
  CopyRoot;
  | Now remove the two independent failure events.
  DeleteEvent = AFW-PUMP-A;
  DeleteEvent = AFW-PUMP-B;
  | Now add the CCF event.
  AddEvent = AFW-PUMP-CCF;
endif
```

The last example demonstrates how a particular cutset could be completely removed from the cutset list. There may be instances in which a cutset should be removed because the combination of basic events should not occur (say for example, two diesel generators out for maintenance at the same time). This rule could be placed in either (or both) the system family rules or the event tree family rules.

Example 4

```
| Define a macro in order to pick up only those cutsets that
| have combinations of two diesel generators out for maintenance.
DIESELS-IN-MAINT = DG-A-MAINT * DG-B-MAINT
| Search for the maintenance events and then delete cutset.
if DIESELS-IN-MAINT then
  | Delete the cutset.
  DeleteRoot;
endif
```

A time-saving feature of the rule editor is that initiating events, basic events, and recovery events can all be added into the database directly from within the rule editor. To add one of these events, simply move the cursor over the name of the event and press <Alt>I to make the event an initiator, <Alt>E to make the event a regular basic event, or <Alt>R to make the event a recovery event. Note that the nomenclature <Alt>KEY signifies that the user press and hold down the "Alt" key while pressing "KEY".

In summary, the recovery rules are a new feature to SAPHIRE 5.0. These rules provide a powerful rule-based method to modify system or sequence cutsets. The keywords and symbols for the rules were defined in Table A-3. The examples presented above suggest the potential applications that can be performed using the SAPHIRE rules.

Appendix B

General MAR-D Data Interchange Formats

Appendix B

General MAR-D Data Interchange Formats

B-1. GENERAL MAR-D FORMAT RULES DATA INTERCHANGE FORMATS—OCTOBER 26, 1994

B-1.1 General Format Rules

1. All name references (family names, event names, etc.) must be upper-case alphanumeric. All lower-case characters will be converted to upper-case. Any alpha fields that are longer than the format specified will be truncated. No spaces are allowed in the middle of names.

2. Descriptions can have both upper-case and lower-case characters. No character checking will be done. No commas are allowed in the description.

3. Commas are used as field delimiters in most formats and can be used as placeholders for unknown fields. Any number of leading and trailing field spaces can be inserted. Exceptions to this format are detailed as needed.

4. Text rules:

    1. File is standard ASCII text, single spaced, upper- and lower-case.

    2. First line of paragraph is indented 5 spaces, with a blank line between paragraphs.

    3. ^EOS signals the End of Section so that multiple names in the same family can be collected in one file.

These rules apply to all files unless specifically stated otherwise.

B-1.2 Family (Plant) Information

B-1.2.1 Family Names and Descriptions

File Name:
xxxxxx.FAD

File Format:
name,description

where

name	- 16 character	Family name (first 8 characters must be unique)
description	- 60 character	Family description

B-1.2.2 Family Attribute File

File Name:
xxxxxx.FAA

File Format:
family=
name,mission,newSum,co,loc,type,design,vendor,AE,OpDate,QualDate

where

name	- 16 character	Family name
mission	- Floating point	Default mission time in hours
newSum	- Floating point	New sequence frequency sum
co	- 10 character	Company name
loc	- 16 character	Location name
type	- 3 character	Facility type
design	- 10 character	Facility design
vendor	- 5 character	Vendor name
AE	- 10 character	Architectural Engineer
OpDate	- (yyyy/mm/dd)	Operational date
QualDate	- (yyyy/mm/dd)	Qualification date

B-1.2.3 Family Textual Information

File Name:
xxxxxx.FAT

File Format:
family =
-- text --

where

family	- 16 character	Family name

B-1.2.4 Family Recovery Rules

File Name:
xxxxxxxx.FAY

File Format:
family =
-- recovery rule text --

where

family	- 16 character	Family name

B-1.2.5 Family Partition Rules

File Name:
xxxxxxxx.FAP

File Format:
family =
-- partition rule text --

where

family	- 16 character	Family name

B-1.3 Basic Event Information

B-1.3.1 Basic Event Names and Descriptions

File Name:
xxxxxx.BED

File Format:
family =
name,description
. . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Event name
description	- 60 character	Alphanumeric description

B-1.3.2 Basic Event Rate Information

The basic event failure rates are stored in the Event relation. This information can also be entered through the Modify option.

File Name:
xxxxxx.BEI

File Format:
family =
name, calc, udC, udT, udV, prob, lambda, tau, mission, init, Flag, udV2
. . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Basic event name
calc	- 1 character	Calculation type

1 - Probability
2 - Lambda * Mission Time
3 - 1 - Exp(-Lambda * Mission Time)
4 - Lambda * Min(Mission Time, Tau)
5 - Operating component with full repair
6 - Lambda * Tau / 2.0
7 - 1+(EXP(-Lambda*Tau)-1.0)/(Lambda*Tau)
8 - Base Probability * Probability
9 - Base Probability * Probability
T - Set to House Event (Failed, Prob=1.0)
F - Set to House Event (Successful, Prob=0.0)
I - Set to ignore
S - Use system mincut upperbound
G - Seismic event-Enter g level for screening
M - Seismic event-Use medium site hazard curve for screening

udC - 4 character Uncertainty correlation class

Events in same class are 100% correlated.

udT - 1 character Uncertainty distribution type

L- Log normal, error factor
N- Normal, standard deviation
B- Beta, b of Beta(a,b)
G- Gamma, a Gamma(a)
C- Chi-squared, degrees of freedom
E- Exponential, none
U- Uniform, Upper end point
H- Histogram
M- Maximum entropy

udV - Floating point Uncertainty distribution value
prob - Floating point Probability value
lambda -Floating point Basic event failure rate per hour
tau - Floating point Time to repair in hours
mission - Floating point Mission time
init - Boolean Initiating event flag (Y/N)
Flag - 1-character process flag
udV2 - Floating point Uncertainty distribution value #2

General Rules:

1. The name field is mandatory.

B-1.3.3 Basic Event Attribute Codes

Basic event attributes are entered through MODIFY--Basic Event and stored in Event.

File Name:
xxxxxx.BEA

File Format:
family =
name,Aname,type,sys,fail,loc,compID,Gname,train,att1,..,att16
. . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Event name
Aname	- 16 character	Alternate event name
type	- 3 character	Event component type
sys	- 3 character	Event component system
fail	- 3 character	Failure mode
loc	- 3 character	Component location
compID	- 7 character	Component ID
Gname	- 16 character	Event group identifier
train	- 3 character	Train identifier
att1..att16	- Class attribute flags--16 values of Y or N (yes or no) indicate whether the attribute described in the class attribute file is applicable.	

General Rules:

1. The name field is mandatory.

B-1.3.4 Basic Event Transformations

Basic event attributes are entered through MODIFY--Basic Event and stored in Event.

Site Name:
xxxxxx.BET

File Format:
family =
name1,level,type
bename1, bename2, . . . ,
. . . , benameN
^EOS
name2,level,type
bename1, bename2, . . . ,
. . . , benameN
^EOS

where

family	- 16 character	Family name
name	- 16 character	Event name
type	- 4 character	Transformation type
level	- 3 character	Transformation level
bename1..N	- 16 character	Event name

B-1.4 Event Attribute Descriptions

B-1.4.1 Failure Mode Descriptions

File Name:
xxxxxx.FMD

File Format:
family =
fail,description
. . . , . . .

where

family	- 16 character	Family name
fail	- 3 character	Failure mode identifier
description	- 60 character	Failure mode description

B-1.4.2 Component Type Descriptions

File Name:
xxxxxx.CTD

File Format:
family =
comp, description
. . . , . . .

where

family	- 16 character	Family name
comp	- 3 character	Component type identifier
description	- 60 character	Component type description

B-1.4.3 System Type Descriptions

File Name:
xxxxxx.STD

File Format:
family =
sys,description
. . . , . . .

where

family	- 16 character	Family name
sys	- 3 character	Component system identifier
description	- 60 character	System description

B-1.4.4 Location Descriptions

File Name:
xxxxxx.LCD

File Format:
family =
loc,description
. . . , . . .

where

family	- 16 character	Family name
loc	- 3 character	Component location identifier
description	- 60 character	Component location description

B-1.4.5 Class Attribute Descriptions

File Name:
xxxxxx.CAD

File Format:
family =
Attr#,description
. . . , . . .

where

family	- 16 character	Family name
Attr#	- Integer 1..16	Attribute number
description	- 60 character	Attribute description

B-1.5 Fault Tree Information

B-1.5.1 Fault Tree Names and Descriptions

File Name:
xxxxxx.FTD

File Format:

family =
name,description[,s]
. . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Fault tree name
description	- 60 character	Fault tree description
s	- 1 character	If included, indicates fault tree is a subsystem

B-1.5.2 Fault Tree Graphics

Fault tree graphics are stored in the block data file of the Graphics relation. The MAR-D file (.DLS) is a display list sequence for the graphics in a binary format. It is loaded and output as is with no conversion performed.

File Name:

xxxxxx.DLS

File Format:

IRRAS 2.5/4.0/5.0 Fault Tree Graphics file (DLS format)

B-1.5.3 Fault Tree Logic

Fault tree logic is stored in the block data file of the Graphics relation.

File Name:

xxxxxx.FTL

File Format:

family, fault tree =
* gatename1,description
gatename1 gatetype input1 input2 . . . inputn
. . ..
* gatenamen,description
gatenamen gatetype input1 input2 . . . inputn
. . .

where

family	- 16 character	Family name
fault tree	- 16 character	Fault tree name
gatename	- 16 character	Gate name
gatetype	- 4 character	Gate type

AND = logical AND
OR = logical OR
TBL = table of events
TRAN = transfer followed by a 16-character fault tree name
NAND = logical NOT AND
NOR = logic NOT OR
N/M = N out of M logic gate
CONT = continuation of inputs to the previous gate

input	- 16 character	Inputs to the gate (event or gate names)
description	- 60 character	Gate name descriptions included as comment

General Rules:

1. A gate definition cannot exceed 255 characters.

2. A line beginning with an asterisk (*) is a comment.

3. For each gate name, a comment should be included giving the gate description.

B-1.5.4 Fault Tree Cut Sets

The fault tree cut sets are stored in the System relation in the block data file.

File Name:
xxxxx.FTC

File Format:

```
family, fault tree, analysis =
eventname * eventname +
eventname * eventname * eventname *
eventname +
eventname * eventname.
^EOS
family, fault tree2 =
```

where

family	- 16 character	Family name
fault tree	- 16 character	Fault tree name
analysis	- 1 character	Analysis type

1 - Random
2 - Fire
3 - Flood

4 - Seismic
5 through 8 - Reserved
9 through 16 - user-defined

eventname	- 16 character	Event names in the cutset

General Rules:

1. An asterisk (*) separates cutset events. Spaces are ignored.
2. A plus sign (+) separates cutsets.
3. A period (.) denotes the end of a sequence.
4. A slash (/) precedes complemented events.
5. Event names are a maximum of 16 characters including the "/".
6. A line beginning with an asterisk (*) is a comment.

B-1.5.5 Fault Tree Attributes

File Name:
xxxxx.FTA

File Format:
family, analysis =
name,level,mission,mincut,proCut,sample,seed,sizCut,sys,cuts,
events,value1,..,value9
. . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . , . . . ,

where

family	- 16 character	Family name
analysis	- 1 character	Analysis type

1 - Random
2 - Fire
3 - Flood
4 - Seismic
5 through 8 - Reserved
9 through 16 - user-defined

name	- 16 character	Fault tree name
level	- Integer 2	0 = top level tree
mission	- Floating point	Mission time
mincut	- Floating point	Mincut upper bound

proCut	- Floating point	Probability cut off value
sample	- Integer 4	Sample size
seed	- Integer 8	Random number seed
sizecut	- Integer 2	Size cut off value
sys	- 3 character	System identifier
cuts	- Integer 5	Base number of cut sets
events	- Integer 5	Base number of events
value	- Floating point	Base uncertainty values

B-1.5.6 Fault Tree Textual Information

File Name:
xxxxxx.FTT

File Format:
family, fault tree =
-- text --
^EOS
family, fault tree2 =
. . .

where

family	- 16 character	Family name
fault tree	- 16 character	Fault tree name

B-1.5.7 Fault Tree Graphical P&ID.

File Name:
xxxxxxxx.PID

File Format:
IRRAS 4.0/5.0 P&ID Graphics file (PID Format)

B-1.5.8 Fault Tree Recovery Rules

File Name:
xxxxxxxx.FTY

File Format:
family =
-- recovery rule text --

where

family	- 16 character	Family name

B-1.6 Event Tree Information

B-1.6.1 Event Tree Names and Descriptions

File Name:
xxxxxx.ETD

File Format:
family =
name,description[,s]
. . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Event tree name
description	- 60 character	Event tree description
s	- 1 character	If included, indicates event tree is a system

B-1.6.2 Event Tree Attributes

File Name:
xxxxxx.ETA

File Format:
family =
name,init
. . . , . . .

where

family	- 16 character	Family name
name	- 16 character	Event tree name
init	- 16 character	Initiating event name

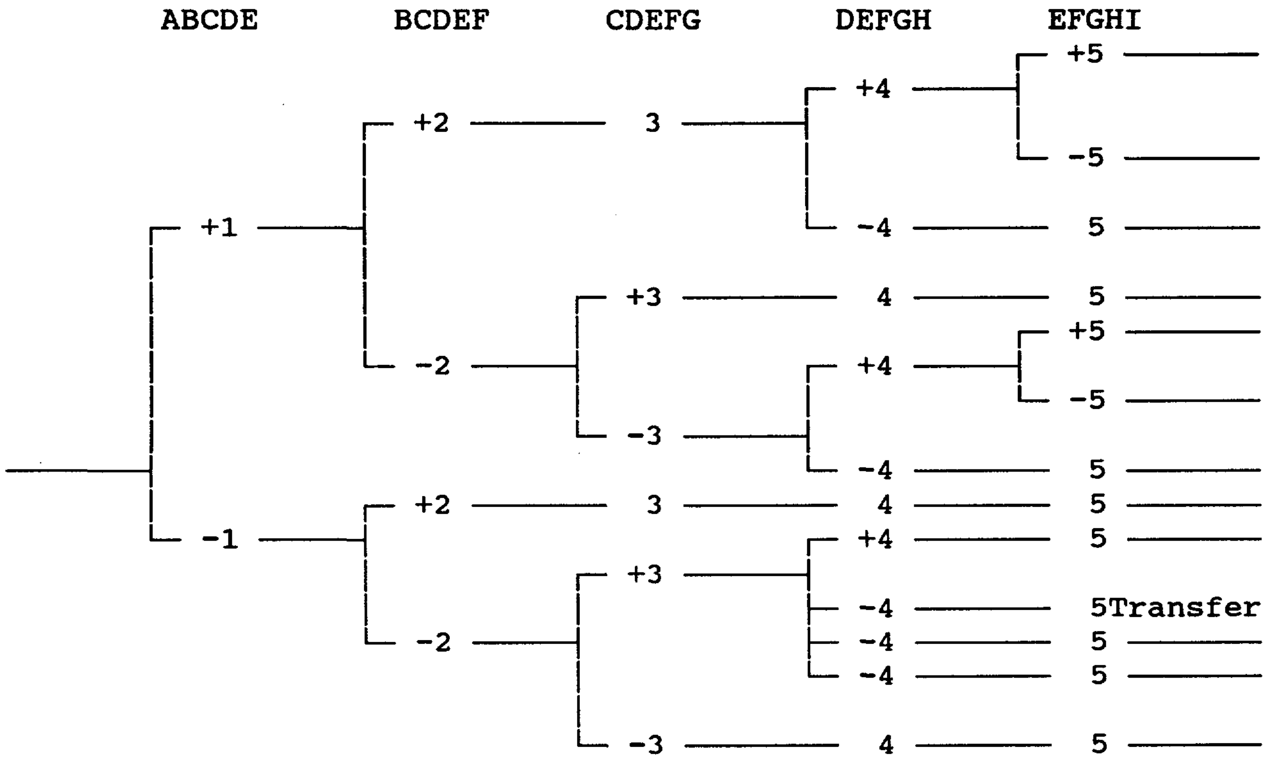
B-1.6.3 Event Tree Graphics

The IRRAS Event Tree Graphics file (*.ETG) is a display list sequence for the graphics. Its format and contents are the same as the Event Tree Logic file.

File Name:
xxxxxx.ETG

File Format:
See file format for the Event Tree Logic

SAMPLE GRAPHICAL EVENT TREE



B-1.6.4 Event Tree Logic

File Name:

xxxxxx.ETL

File Format:

```
family, event tree, init event [,T] =
^TOPS
* 1 | 2 | 3 | 4 | 5 | this is a comment
ABCDE BCDEF CDEFG DEFGH EFGHI
^LOGIC
+1+2  3+  4+   5
              -5
          -4   5
   -2  +3  4   5
       -3 +4  +5
              -5
          -4   5
-1 +2   3  4   5
   -2  +3 +4   5
          -4   5
          -4   5
          -4   5
       -3  4   5
```

```
^SEQUENCES
Y/N, header#1,       Y/N, header#2,Y/N, header#3,Y/N,header#4
Y/N, sequence#1,     Y/N, end state#1,    Y/N, xdata1#1,     Y/N,xdata2#1
Y/N, sequence#2,     Y/N, end state#2,    Y/N, xdata1#2,     Y/N,xdata2#2
Y/N, sequence#3,     Y/N, end state#3,    Y/N, xdata1#3,     Y/N,xdata2#3
Y/N, sequence#4,     Y/N, end state#4,    Y/N, xdata1#4,     Y/N,xdata2#4
Y/N, sequence#5,     Y/N, end state#5,    Y/N, xdata1#5,     Y/N,xdata2#5
Y/N, sequence#6,     Y/N, end state#6,    Y/N, xdata1#6,     Y/N,xdata2#6
Y/N, sequence#7,     Y/N, end state#7,    Y/N, xdata1#7,     Y/N,xdata2#7
Y/N, sequence#8,     Y/N, end state#8,    Y/N, xdata1#8,     Y/N,xdata2#8
Y/N, sequence#9,     Y/N, tran file#9,    Y/N, xdata1#9,     Y/N,xdata2#9, T
Y/N, sequence#10,    Y/N, end state#10,   Y/N, xdata1#10,    Y/N,xdata2#10
Y/N, sequence#11,    Y/N, end state#11,   Y/N, xdata1#11,    Y/N,xdata2#11
Y/N, sequence#12,    Y/N, end state#12,   Y/N, xdata1#12,    Y/N,xdata2#12
Y/N, sequence#13,    Y/N, end state#13,   Y/N, xdata1#13,    Y/N,xdata2#13

^TEXT
SIZE s
JUST j
COLOR j
XY xvalue,yvalue
 "60 character line of text"
XY xvalue, yvalue
 "60 character line of text"
 "60 character line of text"

^PARMS
START yvalue
WINDOW x1,y1,x2,y2
HEADER x1,x2,x3,x4
^EOS
family, event tree2 =
(additional event trees)
```

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
init event	- 16 character	Initiating Event
[,T]	- 1 character	Optional flag indicating init event name is a Top event system
TOPS	- 16 character	Top event/system names
Y/N	- Boolean	End state text displayed?
header	- 16 character	Sequence header
sequence	- 16 character	Sequence name
endstate	- 16 character	End state name
tran file	- 16 character	Name of transfer file

xdata1	- 16 character	Information (optional)
xdata2	- 16 character	Information (optional)

General Rules:

1. A line beginning with an asterisk (*) is a comment.

2. Literal "^TOPS", "^LOGIC", "^SEQUENCES" labels must be present.

3. Logic is built according to the position of the top event in the definition.
   Plus sign (+)---the specified top event succeeded.
   Minus sign (-)---the specified top event failed.
   Blank (   )---the response of the indicated top event did not matter.

4. Header, Sequence name, End State name, Xdata1, Xdata fields associated with each sequence. "Y/N" indicates whether the specified field is visible. A "T" at the end indicates the sequence transfers to another tree.

5. User text is input following the ^TEXT command. Parameters include the size, justification, color, and location of the text block.

6. The ^PARMS command allows input of program control parameters.

B-1.6.5 Event Tree Rules

File Name:
xxxxxxxx.ETR

File Format:
family, event tree =
-- event tree rule text
. . .
^EOS
family, event tree2

where:

family	- 16 character	Family name
event tree	- 16 character	Event tree name
tops	- 16 character	Top event/system names

B-1.6.6 Event Tree Textual Information

File Name:
xxxxxx.ETT

File Format:
family, event tree =
-- text --
^EOS
family, event tree2 =
-- text --

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name

B-1.6.7 Event Tree Recovery Rules

File Name:
xxxxxxxx.ETY

File Format:
family, event tree =
-- recovery rule text --
^EOS
family, event tree2 =

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name

B-1.6.8 Event Tree Partition Rules

File Name:
xxxxxxxx.ETP

File Format:
family, event tree =
-- partition rule text --
^EOS
family, event tree2 =

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name

B-1.7 End State Information

Each sequence can be tied to a single plant damage state. The cutsets for a sequence can be partitioned to map to separate end state. The name and description data are loaded with the SARA *.PDS file.

B-1.7.1 End State Names and Descriptions

File Name:
xxxxxx.ESD

File Format:
family =
name,description
. . . , . . .

where

family	- 16 character	Family name
name	- 16 character	End state name
description	- 60 character	End state description

B-1.7.2 End State Information

File Name:
xxxxxx.ESI

File Format:
family =
****************** will be defined later **************

B-1.7.3 End State Textual Information

File Name:
xxxxxx.EST

File Format:
family, end state =
-- text --
^EOS
family, end state2 =

where

family	- 16 character	Family name
end state	- 16 character	End State name

B-1.7.4 End State Cutsets

The end state cutsets are the minimal cutsets for end state logic as derived from the fault tree logic. The cutsets are stored in the block data file of the ENdState relation.

The MAR-D end state cutsets are in a format similar to that of the fault tree cutsets described in Section 5.1.5.

File Name:
xxxxxx.ENC

File Format:
family, event tree, end state =
eventname * eventname +
eventname * eventname * eventname *
eventname +
eventname * eventname.
^EOS
family, event tree2, end state =

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
end state	- 16 character	End state name
eventname	- 16 character	Event names in the cutset

General Rules:

1. An asterisk (*) separates events in a cutset. Spaces are ignored.
2. A plus sign (+) separates cutsets.
3. A period (.) denotes the end of the sequence.
4. A slash (/) precedes complemented events.
5. Event names have a maximum of 16 characters including the "/" character for complemented events.
6. A line beginning with an asterisk (*) is a comment.

B-1.8 Sequence Information

B-1.8.1 Sequence Names and Descriptions

File Name:
xxxxxx.SQD

File Format:
family,eventree =
name,description
. . . , . . .
^EOS

where

family	- 16 character	Family name
eventree	- 16 character	Event tree name
name	- 16 character	Sequence name
description	- 60 character	Sequence description

B-1.8.2 Sequence Cutsets

The sequence cutsets are the minimal cutsets for sequence logic as derived from the fault tree logic. The cutsets are stored in the block data file of the Sequence relation.

The MAR-D sequence cutsets (.SQC) are in a format similar to that of the fault tree cutsets described in Section 5.1.5.

File Name:

xxxxxx.SQC

File Format:

```
family, event tree, sequence, analysis =
eventname * eventname +
eventname * eventname * eventname *
eventname +
eventname * eventname.
^EOS
family, event tree2, sequence2 =
```

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
sequence	- 16 character	Sequence name
analysis	- 1 character	Analysis type

1 - Random
2 - Fire
3 - Flood
4 - Seismic
5 through 8 - Reserved
9 through 16 - user-defined

eventname	- 16 character	Event names in the cut set

General Rules:

1. An asterisk (*) separates events in a cutset. Spaces are ignored.

2. A plus sign (+) separates cutsets.

3. A period (.) denotes the end of the sequence.

4. A slash (/) precedes complemented events.

5. Event names have a maximum of 16 characters including the "/" character for complemented events.

6. A line beginning with an asterisk (*) is a comment.

B-1.8.3 Sequence Attributes

File Name:

xxxxxx.SQA

File Format:

```
family, event tree, analysis =
name,endstate,mincut,mission,procut,sample,seed,size,cuts,
events,value1, . . . ,value9,default flags, used flags
. . . , . . . , . . . , . . . , . . . , . . . , . . . , . . .
^EOS
family, event tree2 =
```

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
analysis	- 1 character	Analysis type

1 - Random
2 - Fire
3 - Flood
4 - Seismic
5 through 8 - Reserved
9 through 16 - user-defined

name	- 16 character	Sequence name
endstate	- 16 character	End State name
mincut	- Floating point	Mincut upper bound
mission	- Floating point	Mission time in hours
procut	- Floating point	Probability cut off value
sample	- Integer 4	Sample size
seed	- Integer 8	Random number seed
size	- Integer 2	Size cut off value
cuts	- Integer 5	Base number of cutsets
events	- Integer 5	Base number of events
value	- Floating point	Base uncertainty values

value1 - 5th percentile
value2 - Median
value3 - Mean
value4 - 95th percentile
value5 - Minimum sample
value6 - Maximum sample
value7 - Standard deviation
value8 - Skewness
value9 - Kurtosis

Default flags - Default flag set for this sequence
Used flags - Flag set used to generate these cutsets

B-1.8.4 Sequence Logic

File Name:
xxxxxxxx.SQL

File Format:
family, event tree, sequence=
sys1 sys2 /sys3 sys4
. . .
^EOS
family, event tree2, sequence2=

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
sequence	- 16 character	Sequence name
sys	- 16 character	System name

General Rules:

1. Complemented systems are prefixed with "/".

B-1.8.5 Sequence Textual Information

File Name:
xxxxxx.SQT

File Format:
family, event tree, sequence=
--- text ---
^EOS
family, event tree2, sequence2=
--- text ---

where

family	- 16 character	Family name
sequence	- 16 character	Sequence name
event tree	- 16 character	Event tree name

B-1.8.6 Sequence Recovery Rules

File Name:
xxxxxxxx.SQY

File Format:
family, event tree, sequence =
-- recovery rule text --
^EOS
family, event tree, sequence2 =

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
sequence	- 16 character	Sequence name

B-1.8.7 Sequence Partition Rules

File Name:
xxxxxxxx.SQP

File Format:
family, event tree, sequence =
-- partition rule text --
^EOS
family, event tree, sequence2 =

where

family	- 16 character	Family name
event tree	- 16 character	Event tree name
sequence	- 16 character	Sequence name

B-1.9 Piping and Instrumentation Diagrams

B-1.9.1 P&ID

A piping and instrumentation diagram is a graphics file in binary format. It will be loaded and output as-is: no conversion will be performed.

File Name:
xxxxxx.PID

File Format:
(P&ID Editor format)

B-1.10 Gate

B-1.10.1 Gate Description

File Name:
xxxxxx.GTD

File Format:
family=
name,description

where

family	- 16 character	Family name
name	- 16 character	Gate name
description	- 60 character	Gate description

B-1.10.2 Gate Attributes

File Name:
xxxxxx.GTA

File Format
family=
name,attribute

where

family	- 16 character	Family name
name	- 16 character	Gate name
attribute	- 4 character	Gate type

B-1.11 Change Sets

B-1.11.1 Change Set Description

File Name:
xxxxxx.CSD

File Format:
family=
name,description
...,...

where

family	- 16 character	Family name
name	- 16 character	Change set name
description	- 60 character	Change set description

B-1.11.2 Change Set Information

File Name:

xxxxxx.CSI

File Format:

```
family,change=
^PROBABILITY
eventname,calc,udT,prob,lambda,tau,udV,udC,mission,init
^CLASS
eventname,group,compType,compId,system,location,failMode,train,init,att1,..att16
calcType,udT,prob,lambda,tau,udV,udC,mission,init
^EOS
family,change2=
```

where

change	- 16 character	Change set name
eventname	- 16 character	Name mask
group	- 16 character	Event group mask
compType	- 7 character	Component type mask
compId	- 3 character	Component ID mask
system	- 3 character	System mask
location	- 3 character	Location mask
failMode	- 2 character	Failure mode mask
train	- 2 character	Train mask
init	- 1 character	Initiating event (Y/N)
att1..att16	- Class attribute flats--16 values of Y or N (yes or no) indicate whether the attribute described in the class attribute file is applicable.	
calc	- 1 character	Calculation type

1 - Probability
2 - Lambda * Mission Time
3 - 1 - Exp(-Lambda * Mission Time)
4 - Lambda * Min(Mission Time, Tau)
5 - Operating component with full repair
6 - Lambda * Tau / 2.0
7 - 1+(EXP(-Lambda*Tau)-1.0)/(Lambda*Tau)
8 - Base Probability * Probability
9 - Base Probability * Probability
T - Set to House Event (Failed, Prob=1.0)
F - Set to House Event (Successful,Prob=0.0)

I - Set to ignore
S - Use system mincut upperbound
G - Seismic event
L - Use low site hazard curve
M - Use medium site hazard curve
H - Use high site hazard curve

udT	-1 character	Uncertainty distribution type

L-Log normal, error factor
N-Normal, standard deviation
B-Beta, b of Beta(a,b)
G-Gamma, a Gamma(a)
C-Chi-squared, degrees of freedom
E-Exponential, none
U-Uniform, Upper end pt.
H-Histogram
M-Maximum entropy

prob	-Floating point	Probability value
lambda	-Floating point	Basic event failure rate per hr.
tau	-Floating point	Time to repair in hours
udV	-Floating point	Uncertainty distribution value
udC	-4 character	Uncertainty correlation class Events in same class are 100% correlated.
mission	-Floating point	Mission time
init	-Boolean (T/F)	Initiating event

B-1.12 Histograms

B-1.12.1 Histogram Description

File Name:
xxxxxxxx.HID

File Format:
family =
name,type,subtype,description

where

family	- 16 character	Family name
name	- 16 character	Histogram name
type	- 1 character	Histogram type

H - Hazard
U - Uncertainty
F - Fragility

subtype	- 1 character	Histogram subtype

P - Percent
A - Area
R - Range

B-1.12.2 Histogram Information

File Name:
xxxxxxxx.SQP

File Format:
family, histogram =
type, subtype
bin1 value1, bin1 value2
bin1 value2, bin1 value2
...
bin20 value1, bin20 value2
^EOS
family, histogram2 =

where

family	- 16 character	Family name
histogram	- 16 character	Histogram name
type	- 1 character	Histogram type

H - Hazard
U - Uncertainty
F - Fragility

subtype	- 1 character	Histogram subtype

P - Percent
A - Area
R - Range

bin value 1	- Exponential	First value for bin
bin value 2	- Exponential	Second value for bin

B-2. SETS

B-2.1 Sequences

B-2.1.1 Sequence Cutsets

File Name:
xxxxxx.DNF.

The format of the SETS output cutsets file (.DNF) is dependent upon the command issued within SETS. The factored form is

A * (B + C).

The disjunctive normal form is

A * B + A * C.

ONLY the disjunctive normal form is accepted by MAR-D at this time.

File Format:
sequence-name =
eventName * eventName +
eventName * eventName

where
. . . . .

General Rules:

1. An asterisk (*) separates event names. Spaces are ignored.

2. A plus sign (+) separates cutsets.

3. A period (.) denotes the end of a sequence.

4. An asterisk (*) in the first column denotes a comment.

B-2.2 Fault Trees

B-2.2.1 Fault Tree Logic

File Name:
xxxxxx.SET.

File Format:

FAULT TREE$ fault-tree-name.
COMMENT$ descriptive material $
gate-type $ gate-name.IN$ input-1, input-2, . . . , input-n.
OUT$ output-1, output-2, . . . , output-n.
event-type $event-name.OUT$ output-1, . . . , output-n.

where

fault-tree-name - The name of the fault tree.
gate-type - The type of gate being defined.
AG= AND gate
OG= OR gate
EOR= Exclusive OR gate (converted to SG)
EAG= Exclusive AND gate (converted to SG)
SG= Special Gate
gate-name - The name of the gate being defined (16 characters) input-n.
- The names of the gates or primary events that are the immediate inputs to the gate being defined (16 characters).
output-n - The names of the gates that are the immediate outputs of the gate or primary event being defined (16 characters).
event-type - The type of primary event being defined.
BE= Basic Event
CE= Conditional Event
UE= Undeveloped Event
DE= Developed Event
EE= External Event
COMMENT$ - Defines a comment. Must follow a "." delimiter.

B-2.2.2 Fault Tree Cutsets

The fault tree cutsets are stored in the System relation in the block data file. The format of the cutset file (.DNF) is given above.

B-2.3 Basic Event Failure Rates

File Name:

xxxxxxx.VBK.

File Format:

VALUE BLOCK$ value-block-name
prob $ name-list$
prob $ name-list$

where

prob- point value probability estimate
name-list- list of event names separated by commas

B-2.4 Output Reports

The user may convert output reports to other formats or load them into the database. The reports are not stored intact in the database. The cutsets are stripped from the listing file (.LIS) and stored in the Sequence relation. A variable occurrence table is written to file "sequence-name.VOT" in the family directory.

File Name:
xxxxxx.LIS.

File Format:
. . . Header information
EXECUTE
LDBLK (sequence name, sequence name, . . .)
. . .
COMTRMVAL (sequence name)
-- blank line --
/OMEGA means empty cutset
-- 12 blank lines --
Variable Occurrence Table--Output as is
-- 5 lines to cutset table --
41 character leader + 1 space + basic event name

THE MAXIMUM TERM--ends the cutsets
. . .

General Rules:

1. A plus sign (+) followed by a blank line separates cutsets.

2. Cutset terms can be continued on separate lines.

3. An asterisk (*), plus sign (+), or blank ( ) separates basic event names.

4. A period (.) denotes the last cutset.

Appendix C

MAR-D Files For Sample Database

Appendix C

MAR-D Files For Sample Database

C-1. FAMILY FILES

These are examples of files (or partial files) in MAR-D formats for the Sample database. These formats are as of August 1993.

C-1.1 Family Names and Description File (.FAD)

```
SAMPLE        ,This is sample data base
```

C-1.2 Family Attribute File (.FAA)

```
SAMPLE        =
* Name        , Mission   , NewSum    , Company  , Location      ,Typ, Design   ,Vendr, Arch Eng
, OpDate   , QualDate
SAMPLE        , 2.400E+001,------E----,STANDARD  ,HOMETOWN       ,   ,          ,     ,
 ,----/--/--,----/--/--
```

C-1.3 Family Text File (.FAT)

```
SAMPLE        =
     A simple example that models the probability of getting to work on time.
```

C-2. ATTRIBUTES FILES

C-2.1 Train Types File (.TTD)

NO DATA TO EXTRACT

C-2.2 Component Type Description File (.CTD)

NO DATA TO EXTRACT

C-2.3 Failure Mode Description File (.FMD)

NO DATA TO EXTRACT

C-2.4 Location Description File (.LCD)

NO DATA TO EXTRACT

C-2.5 System Type Description File (.STD)

NO DATA TO EXTRACT

C-3. BASIC EVENT FILES

C-3.1 Basic Event Names and Description File (.BED)

SAMPLE =

<FALSE>	,System Generated Success Event
<PASS>	,System Generated Ignore Event
<TRUE>	,System Generated Failure Event
ALARM	,Alarm system fault tree
ALM-BPF	,Alarm fails due to battery failure
ALM-CPF	,Alarm fails due to commercial power failure
ALM-FTS	,Alarm fails because worker fails to set
ALM-MECH	,Alarm fails due to mechanical failure
ALM-SWT	,Alarm fails because worker set wrong time
MEDICINE	,Recovery for sick failure preventing attending work
OTHER	,Other personal reasons that cause a failure to get to work
PER-TRNS	,Personal transportation
PERSONAL	,Personal reasons for failure system fault tree
PUB-TRNS	,Public transportation fails
PUB-TRNS-LAT	,Public transportation fails late time frame
SICK	,Failed to get to work because of illness
SICK-FAM	,Failed to get to work because of illness in family
TRNS-2	,Transportation system fault tree-late time frame
TRNSPRT	,Transportation system fault tree
WORK	,Event tree (WORK) initiating event

C-3.2 Basic Event Rate Information File (.BEI)

SAMPLE =

* Name	,FdT	,UdC	,UdT	,UdValue	, Prob	, Lambda	, Tau	, Mission	,Init
<FALSE>	,1	,	L	1.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	
<PASS>	,1	,	L	1.000E+000	1.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	
<TRUE>	,1	,	L	1.000E+000	1.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	
ALARM	,1	,	L	1.000E+000	1.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	
ALM-BPF	,1	,	L	3.000E+000	9.000E-008	+0.000E+000	+0.000E+000	+0.000E+000	
ALM-CPF	,1	,	L	3.000E+000	1.500E-002	+0.000E+000	+0.000E+000	+0.000E+000	
ALM-FTS	,1	,	L	1.000E+001	5.500E-006	+0.000E+000	+0.000E+000	+0.000E+000	
ALM-MECH	,1	,	L	3.000E+000	2.700E-008	+0.000E+000	+0.000E+000	+0.000E+000	
ALM-SWT	,1	,	L	1.000E+001	2.700E-003	+0.000E+000	+0.000E+000	+0.000E+000	
MEDICINE	,1	,	L	5.000E+000	5.000E-001	+0.000E+000	+0.000E+000	+0.000E+000	R
OTHER	,1	,	L	1.000E+001	8.100E-003	+0.000E+000	+0.000E+000	+0.000E+000	
PER-TRNS	,1	,	L	5.000E+000	5.500E-003	+0.000E+000	+0.000E+000	+0.000E+000	
PERSONAL	,1	,	L	1.000E+000	1.000E+000	+0.000E+000	+0.000E+000	+0.000E+000	
PUB-TRNS	,1	,	L	3.000E+000	2.700E-003	+0.000E+000	+0.000E+000	+0.000E+000	

```
PUB-TRNS-LAT      ,1,   ,L, 3.000E+000, 2.000E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
SICK              ,1,   ,L, 1.000E+001, 8.100E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
SICK-FAM          ,1,   ,L, 1.000E+001, 4.000E-003,+0.000E+000,+0.000E+000,+0.000E+000, ,
TRNS-2            ,1,   ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
TRNSPRT           ,1,   ,L, 1.000E+000, 1.000E+000,+0.000E+000,+0.000E+000,+0.000E+000, ,
WORK              ,1,   ,L, 2.000E+000, 2.480E+002,+0.000E+000,+0.000E+000,+0.000E+000,I,
```

C-3.3 Basic Event Attribute File (.BEA)

```
SAMPLE        =
* Name            ,AltName        ,Typ,Sys,Fail,Loc,CompId, GroupName      ,Train,Attributes
<FALSE>           ,<FALSE>         , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
<PASS>            ,<PASS>          , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
<TRUE>            ,<TRUE>          , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALARM             ,ALARM          ,DE , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALM-BPF           ,ALM-BPF         , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALM-CPF           ,ALM-CPF         , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALM-FTS           ,ALM-FTS         , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALM-MECH          ,ALM-MECH        , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
ALM-SWT           ,ALM-SWT         , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
MEDICINE          ,MEDICING        , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
OTHER             ,OTHER           , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
PER-TRNS          ,PER-TRNS        , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
PERSONAL          ,PERSONAL       ,DE , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
PUB-TRNS          ,PUB-TRNS        , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
PUB-TRNS-LAT      ,PUB-TRNS-LAT    , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
SICK              ,SICK            , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
SICK-FAM          ,SICK-FAM        , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
TRNS-2            ,TRNS-2         ,DE , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
TRNSPRT           ,TRNSPRT        ,DE , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
WORK              ,WORK            , , , , ,    ,              ,  ,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N,N
```

C-4. SYSTEM FILES

C-4.1 Fault Tree Names and Description File (.FTD)

```
SAMPLE        =
ALARM          ,ALARM CLOCK FAILURE
PERSONAL       ,PERSONAL PROBLEMS
TRNS-2         ,COMMERCIAL TRANSPORTATION FAILS AT A LATER TIME
TRNSPRT        ,PERSONAL AND COMMERCIAL TRANSPORTATION FAIL
```

C-4.2 Fault Tree Logic File (.FTL)

```
SAMPLE, ALARM =
ALARM          OR   ALARM-1 ALARM-2 ALM-MECH
ALARM-1        OR   ALM-SWT ALM-FTS
ALARM-2        AND  ALM-CPF ALM-BPF
^EOS
SAMPLE, PERSONAL =
```

```
PERSONAL          OR   SICK SICK-FAM OTHER
^EOS
SAMPLE, TRNS-2 =
TRNS-2            AND  PER-TRNS PUB-TRNS-LAT
^EOS
SAMPLE, TRNSPRT =
TRNSPRT           AND  PER-TRNS PUB-TRNS
```

C-4.3 Fault Tree Graphics File (.DLS)

NOT IN ASCII FORMAT

C-4.4 Fault Tree Cut Sets File (.FTC)

```
SAMPLE, ALARM, 0001=
ALM-FTS +
ALM-MECH +
ALM-SWT +
ALM-BPF * ALM-CPF .
^EOS
SAMPLE, PERSONAL, 0001=
OTHER +
SICK +
SICK-FAM .
^EOS
SAMPLE, TRNS-2, 0001=
PER-TRNS * PUB-TRNS-LAT .
^EOS
SAMPLE, TRNSPRT, 0001=
PER-TRNS * PUB-TRNS .
```

C-4.5 Fault Tree Attribute File (.FTA)

```
SAMPLE, 0001 =
* Name        , Level, Mission   , MinCut    , ProCut    ,Sample, Seed, Siz, Sys, Cuts,Events, UdValues
ALARM       ,0,------E----,2.705E-003,------E----,1000,38547,--,  ,   4,   5, 9.971E-005, 1.157E-003,
2.932E-003, 1.033E-002, 1.714E-005, 1.205E-001, 7.036E-003, 9.028E+000, 1.157E+002
PERSONAL    ,0,------E----, 2.007E-002,------E----, 1000,38547,--,  ,   3,   3, 2.767E-003,
1.216E-002, 2.007E-002, 6.382E-002, 5.473E-004, 3.664E-001, 2.630E-002, 4.975E+000, 4.444E+001
TRNS-2      ,0,------E----, 1.100E-005,------E----,1000,38547,--,  ,   1,   2, 7.866E-007, 5.216E-006,
1.141E-005, 3.744E-005, 2.740E-008, 2.621E-004, 2.098E-005, 6.101E+000, 5.301E+001
TRNSPRT     ,0,------E----,1.485E-005,------E----,1000,46687,--,  ,   1,   2, 1.141E-006, 7.385E-006,
1.504E-005, 5.289E-005, 3.449E-008, 4.646E-004, 2.968E-005, 8.712E+000, 1.133E+002
```

C-4.6 Fault Tree Text File (.FTT)

SAMPLE, ALARM=
The ALARM fault tree is a simple representation modeling alarm clock failure. Some common reasons for alarm clock failure include setting the wrong time, mechanical failure, or power failure (either battery or commercial).

C-5. EVENT TREE FILES

C-5.1 Event Tree Names and Descriptions File (.ETD)

```
SAMPLE          =
WORK            ,WORK EVENT TREE
```

C-5.2 Event Tree Graphics File (.ETG)

```
SAMPLE, WORK, WORK =
^TOPS
ALARM, PERSONAL, TRNSPRT
^LOGIC
 +1 +2 +3
       -3
    -2  3
 -1  2 +3
       -3
^SEQUENCES
Y, SEQUENCE-NAMES, Y, END-STATE-NAMES, N, FREQUENCY, N, EXTRA-#1,
Y, A, Y, OK, Y, , Y, ,
Y, B, Y, LATE-TO-WORK, Y, , Y, ,
Y, C, Y, MISS-WORK, Y, , Y, ,
Y, D, Y, LATE-TO-WORK, Y, , Y, ,
Y, E, Y, LATE-TO-WORK, Y, , Y, ,
^TOPDESC
 "INITIATING EVENT"
!
 "ALARM FAILURE"
!
 "PERSONAL FAILURE"
!
 "TRANSPORTATION",
 "FAILURE"
!
^PARMS
START 80.96 WINDOW 17.00, -3.11, 147.00, 126.89
HEADER 97.20, 122.40, 147.60, 172.80
STRING E
```

DEFFONT 5
TOPWIDTH 9
TOPSIZE 1.44
TOPFONT 1
DESHITE 3
DESSIZE 1.00
DESFONT 5
NODEHITE 2.00
ENDSIZE 1.50
ENDFONT 1

C-5.3 Event Tree Logic File (.ETL)

SAME AS THE .ETG FILE SECTION C.5.2

C-5.4 Event Tree Attribute File (.ETA)

```
SAMPLE         =
*  Name        , Init Event
WORK            , WORK
```

C-5.5 Event Tree Rules File (.ETR)

```
SAMPLE, WORK=
SET , Rule to substitute TRNS-2 for TRNSPRT
IF   ALARM
THEN TRNSPRT = TRNS-2.
```

C-5.6 Event Tree Text File (.ETT)

SAMPLE, WORK=

A FAIL-SUCCESS LOGIC WAS USED TO DEVELOP AN EVENT TREE TO CALCULATE THE FREQUENCY THAT THE AVERAGE PERSON WILL ARRIVE ON TIME, BE LATE, OR MISS A DAY OF WORK.

C-6. END STATE FILES

C-6.1 End State Names and Description File (.ESD)

```
SAMPLE          =
LATE-TO-WORK    , This end state represents being late to work
MISS-WORK       , This end state represents missing work
```

C-6.2 End State Text File (.EST)

SAMPLE, LATE-TO-WORK=
THIS IS THE LATE TO WORK END STATE.

C-7. SEQUENCE FILES

C-7.1 Sequence Names and Description File (.SQD)

```
SAMPLE, WORK=
2            ,LATE-TO WORK
3            ,MISS-WORK
4            ,LATE-TO-WORK
5            ,LATE-TO-WORK
```

C-7.2 Sequence Cut Set File (.SQC)

```
SAMPLE, WORK, 2, 0001=
PER-TRNS * PUB-TRNS .
^EOS
SAMPLE, WORK, 3, 0001=
OTHER +
SICK * MEDICINE +
SICK-FAM .
^EOS
SAMPLE, WORK, 4, 0001=
ALM-FTS +
ALM-MECH +
ALM-SWT +
ALM-BPF * ALM-CPF .
^EOS
SAMPLE, WORK, 5, 0001=
ALM-FTS * PER-TRNS * PUB-TRNS-LAT +
ALM-MECH * PER-TRNS * PUB-TRNS-LAT +
ALM-SWT * PER-TRNS * PUB-TRNS-LAT +
ALM-BPF * ALM-CPF * PER-TRNS * PUB-TRNS-LAT .
```

C-7.3 Sequence Cutset Attribute File (.SQA)

```
SAMPLE, WORK, 0001=* Name          , End State       , Mission    , MinCut      , ProCut
,Sample,Seed,Siz,Cuts,Events, UdValues, Def Flags, Used Flags
2              ,LATE-TO-WORK     , 3.682E-003,------E----,------E----, 1000,40777,--,      1,     3,
2.336E-004, 1.658E-003, 3.699E-003, 1.262E-002, 5.091E-005, 9.751E-002, 7.072E-003, 6.588E+000,
6.447E+001,
```

```
3              ,MISS-WORK        , 3.985E+000,------E----,------E----, 1000,46267,--,    3,    5,
3.729E-001, 1.959E+000, 4.303E+000, 1.279E+001, 8.912E-002, 1.813E+002, 9.597E+000,
1.032E+001, 1.531E+002,              ,
4              ,LATE-TO-WORK     , 6.709E-001,------E----,------E----, 1000,52257,--,    4,    6,
2.007E-002, 2.348E-001, 7.012E-001, 2.677E+000, 2.867E-003, 1.839E+001, 1.578E+000,
5.623E+000, 4.365E+001,              ,
5              ,LATE-TO-WORK     , 7.380E-006,------E----,------E----, 1000,58407,--,    4,    8,
4.896E-008,1.232E-006,1.171E-005,3.014E-005,1.073E-009,3.977E-003,1.307E-004,2.831E+001,
8.500E+002,          ,
```

C-7.4 Sequence Logic File (.SQL)

```
SAMPLE, WORK, 2=
/ALARM /PERSONAL TRNSPRT .
^EOS
SAMPLE, WORK, 3=
/ALARM PERSONAL .
^EOS
SAMPLE, WORK, 4=
ALARM /TRNS-2 .
^EOS
SAMPLE, WORK, 5=
ALARM TRNS-2 .
```

C-7.5 Sequence Text File (.SQT)

SAMPLE, WORK, 3=

Sequence 3 is the event tree sequence that is used to demonstrate the use of recovery rules or recovery actions.

C-8. GATE FILES

C-8.1 Gate Description File (.GTD)

```
SAMPLE     =
ALARM      , ALARM CLOCK FAILURE
ALARM-1    , ALARM CLOCK SETTING FAILURE
ALARM-2    , ALARM CLOCK POWER FAILURE
PERSONAL   , PERSONAL PROBLEMS
TRNS-2     , COMMERCIAL TRANSPORTATION FAILS AT A LATER TIME
TRNSPRT    , PERSONAL AND COMMERCIAL TRANSPORTATION FAILURE
```

C-8.2 Gate Attributes File (.GTA)

SAMPLE	=
* Name	, Type
ALARM	, OR
ALARM-1	, OR
ALARM-2	, AND
PERSONAL	, OR
TRNS-2	, AND
TRNSPRT	, AND

C-9. CHANGE SETS FILES

C-9.1 Change Sets Description File (.CSD)

NO DATA TO EXTRACT

C-9.2 Change Sets Information File (.CSI)

NO DATA TO EXTRACT

Appendix D

Seismic Data Loading

Appendix D

Seismic Data Loading

D-1. INTRODUCTION

This appendix discusses the features and basic data loading processes of the seismic module in SAPHIRE 5.0. The seismic data loading process assumes the availability of an internal-events PRA or database (i.e. a SAPHIRE database implementing analysis with random failures within a particular system). The procedures necessary for seismic data loading using the SAPHIRE code are described in the following subsections.

D-2. SAPHIRE SEISMIC CAPABILITIES

The SAPHIRE seismic analysis capabilities are designed to function directly from the internal-events PRA. Thus, internal basic events, system fault tree models, accident sequences, and initiating events have all been defined and developed for the system of interest. The SAPHIRE seismic analysis consists of taking the internal basic events (having random failures) and converting them into seismic basic events that represent seismic-induced failures. SAPHIRE performs transformations in the form of boolean identities that allows the user to build on an internal-events analysis when developing a seismic model. After seismic vulnerabilities have been identified, they are incorporated into an existing internal-events analysis using a set of basic event transformations that substitute in seismic-induced failures that are used to generate seismic sequence or system cutsets.

D-3. BUILDING AND LOADING THE SEISMIC SAPHIRE MODEL

D-3.1 Hazard Curves

The hazard curve represents a range of possible earthquake magnitudes. The curve is usually found in the form of a probability of exceedence curve, with the earthquake ground acceleration on the horizontal axis and the probability of exceeding that acceleration on the vertical axis. (Sources of hazard curve data and information include NUREG-1488 and NUREG-4550.) SAPHIRE uses this information in the form of a histogram or a discreet probability density distribution. For a more detailed description of hazard curves and the methodology on their use during seismic analysis, see the SAPHIRE Version 5.0 Technical Reference Manual.[4]

The hazard curve (or histogram) that will be used in the seismic analysis is developed or modified by selecting the desired seismic hazard curve in the IRRAS subprogram. This is done by selecting the "MODIFY Data Base" - "Family" - "Modify" series of menu commands. Under the heading "Site Hazard Curves", there are three fields: "Low", "Medium", and "High". The histogram listed in the "Medium" field will be the one used during analysis. If a seismic hazard curve is not available, then one must be added in order to generate quantified cutsets.

A seismic hazard curve (or histogram) can be added (or loaded) into the SAPHIRE database using two methods. The histogram can be added and the discrete data points input from the "Modify

Data Base" module or it can be loaded from a histogram flat file (.HII) through the MAR-D subprogram. The procedures for both methods are discussed below.

D-3.1.1 Loading the Seismic Histogram Through the Modify Data Base Module

To add a seismic histogram from within the IRRAS program, the following steps are required:

1. Select "Modify Data Base" - "Family".

2. Highlight the Family in which the specific hazard curve is to be implemented. Choose "Modify".

3. Under the heading "Site Hazard Curves", highlight the blank "Medium" block. Press <F1> for a list of available histograms and to add new ones.

4. Press <F9> to add a Histogram. This puts the user in the "Add Histogram" screen.

5. Enter the name and description of the seismic histogram.

6. Enter the acceleration rates and frequencies. The acceleration rate is the peak ground acceleration (i.e., magnitude of the earthquake). The frequency is the probability that an earthquake that exceeds the ground acceleration will occur.

D-3.1.2. Loading the Seismic Histogram Through the MAR-D Interface

The hazard curve (or histogram) may also be loaded into the SAPHIRE database using the MAR-D subprogram. The histogram is represented in an ASCII text file and loaded into the SAPHIRE database using the MAR-D/LOAD/HISTOGRAM option, as discussed in Appendix A. The two flat file types that are required to load the histogram using MAR-D are discussed below.

D-3.1.2.1 Histogram Description File (.HID). The MAR-D flat file format for the histogram description file (.HID) is:

File Name:
xxxxxxxx.HID

File Format:
family=
name, type, subtype, description

where

family	-16 character	Family name
name	-16 character	Histogram name

type	-1 character	Histogram type H - Hazard U - Uncertainty F - Fragility
subtype	-1 character	Histogram subtype P - Percent A - Area R - Range

An example of a histogram description file in MAR-D format is as follows:

```
SAMPLE     =
SEISMIC    , H, H, Histogram for Seismic Analysis
```

D-3.1.2.2 Histogram Information File (.HII). The MAR-D data format for the histogram information file (.HII) is:

File Name:

```
xxxxxxxx.HII
```

File Format:

```
family, histogram =
type, subtype
bin1 value1, bin1 value2
bin2 value1, bin2 value2
...
bin20 value1, bin20 value20
^EOS
```

where

family	- 16 character	Family name
histogram	- 16 character	Histogram name
type	- 1 character	Histogram type H - Hazard U - Uncertainty F - Fragility
subtype	- 1 character	Histogram subtype P - Percent A - Area R - Range
bin1 value1	- Exponential	First value for bin 1
bin1 value2	- Exponential	Second value for bin 1

An example of a histogram information file in MAR-D format is shown below. For this example, the flat file will load seven bins with seismic hazard histogram data. For all .HII files containing seismic data, "bin1 value1" or column 1 is the earthquake frequency (per yr) and "bin1 value2" or column 2 is the mean failure acceleration of the earthquake.

```
SAMPLE, SEISMIC =
H, H
 3.680E-003, 1.000E-001
 2.980E-004, 2.000E-001
 7.200E-005, 3.000E-001
 2.620E-005, 4.000E-001
 1.170E-005, 5.000E-001
 6.000E-006, 6.000E-001
 3.360E-006, 7.000E-001
```

D-3.2 Event Trees

The creation of a seismic analysis model in the IRRAS subprogram requires the development of a seismic event tree. The seismic event tree can be designed to incorporate the seismic analysis by either of two methods. The first method utilizes the internal basic events and fault trees assumed already present in the database. This method prioritizes and links the seismic-induced internal events and fault trees and will generate seismic sequence cutsets from the internal basic events. The second method utilizes separated seismic fault tree logic that may incorporate internal events or separate seismic events to generate the seismic cutsets. For both methods, the seismic event tree begins with a generic seismic-initiating event set to a value of 1.0 (True Event). The actual magnitude and frequency of the earthquake of interest are identified by the user and factored into the analysis when the cutsets are generated and quantified.

The top events for the seismic event tree are those events or systems that have the potential to be induced by an earthquake. They are listed in order of severity, with the more severe-induced initiators listed first. This also addresses the potential pitfall of overcounting core damage sequences where, for example, a single earthquake inducing both a large LOCA and a small LOCA at the same time. During the seismic analysis, the event tree top events are treated as seismic events with the associated seismic fragility data.

The procedures for loading or adding event trees to the SAPHIRE database was discussed in Section 4.3. The same procedures are required for loading the seismic event trees and any related subtrees.

D-3.3 Fault Trees

The seismic system models (i.e., fault trees) can be created in SAPHIRE either as independent, stand-alone seismic fault trees, or they can also be integrated with the internal events analysis. To integrate seismic analysis into the internal events analysis, transformations need to be defined that convert random failures to seismic-induced failures.

Because the internal fault trees do not include several seismic related basic events, the basic events must be either added to the internal fault trees or independent seismic fault trees must be created. The procedures for loading or adding system fault trees was discussed in Section 4.5.

D-3.4 Basic Event Data

In most instances, seismic basic events are transformed internal basic events where the seismic considerations are implemented after the transformations. Seismic failure data are usually characterized by a median fragility and two uncertainty terms representing the random uncertainty and confidence uncertainty (Beta-R and Beta-U, respectively). See the SAPHIRE Technical Reference Manual[4] for a more indepth discussion of seismic fragility and component failure probabilities.

The necessary steps in loading seismic basic events into the SAPHIRE program are:

1. Add the seismic event to the database including any basic event attribute data.

2. Enter the seismic failure acceleration data.

3. Enter the seismic uncertainty data.

4. Modify any internal basic events that are determined to have seismic vulnerabilities to include a seismic susceptibility "flag". This will allow for the internal basic event to be transformed into a new seismic event.

5. Enter the transformation definition to the internal basic event that is seismic susceptible.

These steps are further discussed in the following sections.

D-3.4.1 Adding Seismic Basic Events

Before the internal basic event transformation can be created, the seismic basic events must be defined. In most cases, the newly created seismic event has a different name than the internal basic event name that it is transformed from. For example, if the internal basic event HPI-MOV-FO-108A is determined to be seismic susceptible, then it must be transformed into a seismic event. The new seismic event could be named S-HPIMOV-FO-108A and must be added to the database.

The procedure for adding seismic basic events and their descriptions is identical to that of internal basic events and is discussed in Section 4.6.

D-3.4.2 Loading the Seismic Failure Acceleration Data

Loading of the seismic failure data is similar to the procedures discussed for loading failure data discussed in Section 4.6. Two methods can be used to load seismic failure acceleration data. The data can be entered in the Modify Data Base module or from basic event flat file (.BEI) and loaded through MAR-D as described in Appendix A. Differences between loading seismic data and the procedures discussed in Section 4.6 are outlined below.

D-3.4.2.1 Loading Through the Modify Data Base Module. To enter seismic data into a seismic basic event record, go to the "Failure Data Calculation Type". Enter a "G" or an "H", which defines the basic event as a seismic basic event. Entering a "G" allows the user to input an assumed g-level (earthquake strength) for use in initially generating cutsets. The "H" tells SAPHIRE to use the highest g-level (bin) from the histogram identified for use in the Modify-Family option.

D-3.4.2.2 Loading Through the MAR-D Interface. The loading of seismic failure data through the MAR-D interface is similar to the procedures described in Section 4.6 for loading internal basic event failure rates. The seismic basic event flat file (.BEI) data format is similar to that in Appendix B except for the following:

1. Set the calculation type (calc) to "G" or "H" to define the basic event as a seismic event.

2. Place the Seismic Failure value in the .BEI "prob" position.

3. If a calculation type of "G" is used, specify an earthquake "G-Level". Place it in the .BEI "Lambda" position.

D-3.4.3 Loading the Seismic Uncertainty Data

Loading of the seismic uncertainty data is similar to the procedures discussed in Section 4.6. Two methods can be used to load seismic uncertainty data. The data can be entered in the Modify Data Base module or from a basic event flat file (.BEI) and loaded through MAR-D as described in Appendix A. Differences between loading seismic data and the procedures discussed in Section 4.6 are outlined below.

D-3.4.3.1 Loading Through the Modify Data Base Module. To enter seismic uncertainty data into a seismic basic event record, go to the "Failure Data Calculation Type". Enter an "S", which defines the basic event as a seismic basic event. Enter the Beta-R and the Beta-U in their respective blocks.

D-3.4.3.2 Loading Through the MAR-D Interface. The loading of seismic uncertainty data through the MAR-D interface is similar to the procedures described in Section 4.6 for loading internal basic event uncertainties. The seismic basic event flat file (.BEI) data format is similar to that described in Appendix B except for the following:

1. Set the uncertainty type (UdT) to "S" to allow for the implementation of seismic uncertainties.

2. Specify the seismic uncertainty term representing the random uncertainty, Beta-R. Place this value in the .BEI UdValue position. Specify the confidence uncertainty term, Beta-U, and place it in the .BEI UdValue2 position.

D-3.4.4 Defining Internal Event Susceptibility to Seismic Activity

In order to integrate the internal event analysis with a seismic analysis, the internal basic events must be transformed into new seismic events. This process first involves defining the internal basic events that are as seismically susceptible. Basic event susceptibility can be entered into the SAPHIRE database using either the Modify Data Base module or from a basic event attribute flat file (.BEA) through the MAR-D subprogram interface. Both methods are discussed below.

D-3.4.4.1 Defining Susceptibility Through the Modify Data Base Module. An internal event that is determined to be seismically vulnerable is defined in SAPHIRE as seismically susceptible. This is done in the "Modify Data Base/Basic Events". Highlight the desired internal event and chose "Modify". Under "Susceptibilities", change the option #4 block from "N" to "Y", which will identify this particular basic event as susceptible to seismic initiators.

D-3.4.4.2 Defining Susceptibility Through MAR-D. An internal basic event flat file (.BEA) can be generated from MAR-D as is described in Appendix A. The file format of the .BEA is described in Appendix B. To define a basic event as seismic susceptible, attribute 4 (att4) must be changed from "N" to "Y". Reloading this .BEA file with the seismic susceptible attribute is described in Appendix A.

D-3.4.5 Defining the Internal Basic Event Transformations

A transformation is a replacement or addition inside the fault tree logic. An internal event that is determined to be seismically vulnerable needs to be transformed into a new seismic event in SAPHIRE. During the transformation process, the internal basic event is replaced with a seismic basic event or a series of seismic events.

SAPHIRE utilizes three types of transformations: (1) AND, (2) OR, and (3) ZOR. An "AND" type transformation replaces the event being transformed with an AND gate having any transformed events as inputs. An "OR" type transformation replaces the event being transformed with an OR gate having any transformed events as inputs. An "ZOR" type transformation implies that if any transformed events from the original transformed event fails, then all events fail. Since for seismic analysis, an internal random basic event is transformed into one new seismic basic event, the transformation type should be "OR". This will prevent the random event and the seismic event from being "ANDed" together during the seismic analysis.

Basic event transformation also require a "transformation level" that indicates the level of substitution for the transformation. The transformation is an integer between 0 and 255. The reader is directed to the IRRAS Reference Manual[2] for discussion of transformations.

Transformation data can be entered into the SAPHIRE database using either the Modify Data Base module or from a basic event transformation flat file (.BET) through the MAR-D subprogram interface.

D-3.4.5.1 Loading Seismic Transformations Using the Modify Data Base Module. Basic event transformation is accomplished in SAPHIRE through the "Modify Data Base/Basic Events" module. This is done by with the following steps:

1. Highlight the desired internal event and chose "Transformations". A new screen appears entitled "Transformations". This screen consists of a "Name" and a "Description" block.

2. Enter into the "Name" block the name of the seismic event for which the user wishes to transform the original event. By pressing the "ENTER" key, the entire basic event listing is available and the user may simply mark the desired seismic event and it will appear in the "Name" block.

3. A description of the seismic basic event may be entered in the "Description" block.

4. While in the "Transformation" screen, press the "Esc" key. This screen allows the user to enter the transformation type and level.

5. Press "Enter" to exit "Transformation" module.

D-3.4.5.2 Loading Seismic Transformations with the MAR-D Utility. Basic event transformation may also be loaded into SAPHIRE through a MAR-D file (.BET). Below is the MAR-D file format for the basic event transformation file (.BET).

Site Name:
xxxxxxxx.BET

File Format:
```
family=
name1, level, type
bename1, bename2, ...,
..., benameN
^EOS
```

where

family	- 16 character	Family name
name1	- 16 character	Event name (to be transformed)
level	- 3 character	Transformation level
type	- 4 character	Transformation type
bename1..N	- 16 character	Event name (transformed event)

The loading of a MAR-D flat file into SAPHIRE is described in detail in Appendix A.

D-4. GENERATING AND QUANTIFYING SEISMIC CUTSETS

Generating and quantifying seismic cutsets at both the systems level and the sequence level is similar to that for internal (random) analysis described in Sections 4.5.4 and 4.5.7, respectively. The few minor differences are noted below.

D-4.1 Generating Seismic Cutsets

When generating seismic cutsets during both fault tree and sequence analysis, the user must specify that seismic analysis is desired. This is accomplished in both the "Analyze Systems" and "Analyze Sequences" menu screens of IRRAS. To change from "Random" analysis to "Seismic" analysis, the user will have two options:

1. Access "Utility Options" - "Define Constant" menu. Under "Transformations", change the "Analysis type" block to "Seismic".

2. Access the "Analysis type" option located in both the "Analyze systems" and "Analyze sequences" menus. Change the "Analysis Type" to "Seismic".

D-4.2 Quantifying Seismic Cutsets

When quantifying seismic cutsets during both fault tree and sequence analysis, the user should confirm that the "Analysis type" is set to "Seismic". In addition, after selecting "Quantification", the user is prompted to choose the "G-Level" for which quantification is to be performed. The options available for "G-level" quantification include:

1. Selecting one of the g-level bins that contain a non-zero value obtained from the hazard histogram identified for use with the current Family.

2. Selecting "ALL COMBINED". This gives an overall value obtained by adding the data using all bins in the histogram.

3. Selecting "ALL SEPARATE". This quantifies the cutsets at each g-level bin that contains a non-zero value obtained from the hazard histogram used with the current Family. It should be noted that after quantification using the "ALL SEPARATE" option, the cutset list for each g-level is not maintained. When quantification is completed, only the last quantification performed (at that specific g-level) is available. However, numerical results are stored and are available for each individual g-level that was calculated. These individual results are generally used during uncertainty analysis.

NRC FORM 335 (2-89) NRCM 1102, 3201, 3202

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

(See instructions on the reverse)

1. REPORT NUMBER (Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if any.)

NUREG/CR-6116
INEL-94/0039
Vol. 10

2. TITLE AND SUBTITLE

Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 5.0

Data Loading Manual

3. DATE REPORT PUBLISHED

MONTH	YEAR
April	1995

4. FIN OR GRANT NUMBER

W6241

5. AUTHOR(S)

R. L. VanHorn, L. M. Wolfram, R. D. Fowler, S. T. Beck, C. L. Smith

6. TYPE OF REPORT

Technical

7. PERIOD COVERED (Inclusive Dates)

8. PERFORMING ORGANIZATION - NAME AND ADDRESS (If NRC, provide Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)

Idaho National Engineering Laboratory
Lockheed Idaho Technologies Company
P.O. Box 1625
Idaho Falls, ID 83415

9. SPONSORING ORGANIZATION - NAME AND ADDRESS (If NRC, type "Same as above"; if contractor, provide NRC Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address.)

Division of Systems Technology
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555-0001

10. SUPPLEMENTARY NOTES

11. ABSTRACT (200 words or less)

The Systems Analysis Programs for the Hands-on Integrated Reliability Evaluations (SAPHIRE) suite of programs can be used to organize and standardize in an electronic format information from probabilistic risk assessments or individual plant examinations. The Models and Results Database (MAR-D) program of the SAPHIRE suite serves as the repository for probabilistic risk assessment and individual plant examination data and information. This report demonstrates by examples the common electronic and manual methods used to load these types of data. It is not a stand alone document but references documents that contribute information relative to the data loading process. This document provides a more detailed discussion and instructions for using SAPHIRE 5.0 only when enough information on a specific topic is not provided by another available source.

12. KEY WORDS/DESCRIPTORS (List words or phrases that will assist researchers in locating the report.)

FEP
IRRAS
MAR-D
PRA data loading
SAPHIRE
SARA

13. AVAILABILITY STATEMENT

Unlimited

14. SECURITY CLASSIFICATION

(This Page)
Unclassified

(This Report)
Unclassified

15. NUMBER OF PAGES

16. PRICE