N92-22468

HYBRID AUTOMATED RELIABILITY PREDICTOR INTEGRATED WORK STATION

HIREL

Salvatore J. Bavuso

NASA Langley Research Center

Hampton, VA 23665-5225

ABSTRACT

The Hybrid Automated Reliability Predictor (HARP) integrated reliability (HiREL) work station tool system marks another accomplishment toward the goal of producing a totally integrated computer-aided design (CAD) work station design capability. Since a reliability engineer must generally graphically represent a reliability model before he can solve it, the use of a graphical input description language increases productivity and decreases the incidence of error. The captured image displayed on a cathode ray tube (CRT) screen serves as a documented copy of the model (as a hard copy can be readily made by the push of a button) and provides the data for automatic input to the HARP reliability model solver. The introduction of dependency gates to a fault tree notation allows the modeling of very large fault tolerant system models using a concise and visually recognizable and familiar graphical language. In addition to aiding in the validation of the reliability model, the concise graphical representation presents company management, regulatory agencies, and company customers a means of expressing a complex model that is readily understandable. The graphical postprocessor computer program HARPO (HARP Output) makes it possible for reliability engineers to quickly analyze huge amounts of reliability/availability data to observe trends due to exploratory design changes. HiREL is written in ANSI standard code for maximum portability and has been successfully executed on IBM compatible 286/386/486 personal computers, Sun and Vaxstation platforms. The major components of HiREL have already proven themselves to be a useful modeling asset to a number of aerospace companies that have been serving as beta test sites since 1985.

INTRODUCTION

Electronic design engineers are increasingly faced with shorter design cycle times which account to a large extent for the heightened interest in computer-aided design software (CAD) tools. In conjunction with the advent of affordable powerful work stations, CAD software is becoming a mainstay capability in the engineering community. The success of current CAD tools has encouraged the engineering community to seek a capability that totally integrates the system design process. Although this capability does not presently exist, many of the software components that are required for such a capability are presently available.

One such software component that has recently been developed and released to the engineering community is HiRel: the Hybrid Automated Reliability Predictor (HARP) integrated Reliability system tool for reliability/availability prediction (ref. 1). HiRel offers a toolbox of integrated software modules that can be used to customize the user's application in a work station environment. It consists of two interactive graphical input/output modules and four reliability/availabilty modeling engines that provide analytical and simulative solutions to a wide host of highly reliable fault-tolerant system architectures and is also applicable to electronic systems in general.

The tool system was designed at the outset to be compatible with most computing platforms and operating systems and some modules have been beta tested within the aerospace community for over seven years. Over 100 copies have been distributed. Many examples of its use have been reported in the literature and at the HARP Workshop conducted at Duke University, July 10-11, 1990 (ref. 2).

PRECEDING PAGE BLANK NOT FILMED

HIREL DEVELOPMENT

The development of HiRel has been an evolutionary project that has spanned over nearly two decades. The goal that was set for HiRel circa 1973 was to develop a capability to assess the reliability/availability of any fault-tolerant digital computer-based system, including the system effects of software, i.e., fault/error handling. Although the initial target application was for assessing highly reliable real-time digital fault-tolerant aircraft flight control systems, the developers of HiRel did not limit its applicability to soley that application. The realization that one day NASA spacecraft would require ultrahigh fault-tolerant systems, motivated the HiRel developmental team to include a reliability modeling capability to accurately represent non-constant failure rate models as well as constant failure rate models typically found in aircraft applications. Data to support the use of the decreasing failure rate model were published as early as 1975 and provided much of the motivation (ref. 3). This foresight was fortuitous since there now exists significant data to productively use HiRel's Weibull decreasing failure rate parts model. For very long manned missions, such as a mission to Mars presently under consideration by NASA, decreasing failure rate models may well prove to be the modeling technology that can provide the reliability confidence to make such a trip.

HIREL DESCRIPTION

The wide range of applications of interest has caused HiRel to evolve into a family of independent software modules that communicate with each other through files that each module generates. In this sense, HiRel offers a tool box of integrated software modules that can be executed to customize the user's application. Figure 1 depicts the HiRel tool system. The core of this capability consists of the reliability/availability modeling engines, which are collectively called the Hybrid Automated Reliability Predictor (HARP). It is comprised of four self-contained executable software components: The original HARP module, Monte Carlo HARP (MC-HARP), Phased Mission HARP (PM-HARP), and X -Windows HARP (XHARP). In conjunction with the engine suite, there are two input/output interactive Graphical User Interface (GUI) modules that provide a work station environment for HiRel. These software modules are called the Graphics Oriented (GO) module and the HARP Output (HARPO) module.

Reliability/Availability Modeling Engines

The power of HiRel resides in its engine suite which consumed the bulk of the development effort and took over a decade to complete. A mathematical modeling technique called behavioral decomposition and a fault tree notation called dynamic fault trees constitute the major engine suite modeling power which is used by the other engine suite modules. The engine suite is composed of independently executable software modules: HARP/behavioral decomposition, HARP/dynamic fault tree model, MC-HARP, PM-HARP, and X-HARP. Since the original HARP is the kernel of the engine suite, it will be discussed first.

HARP/Behavioral Decomposition Modeling Engine

The prototype reliability/availability engine was implemented into the "Textual" HARP software subsystem which initially was a stand-alone system that later became integrated into HiRel. Textual HARP offers a textual interactive input capability when executed in a stand-alone mode and will execute on many computing platforms requiring only an ANSI standard FORTRAN compiler. The GO software module is offered as a complementary input capability to the textual input format but requires the installation of a commercially available graphical software package for execution.

HARP is a software tool for analytically predicting the reliability/availability of fault-tolerant digital computer-based systems; however, it is also applicable to a very large class of systems in general. In addition to reliability/availability, it can be used to analyze system sensitivity and failure causes. Its notable features include: very large system modeling, dynamic fault modeling, automatic conversion of fault tree input to a Markov chain or manual Markov chain input, automatic insertion of fault handling models into Markov chains, automatic parametric analysis, and wide portability of the code. It utilizes a method called behavioral decomposition to solve for the reliability of a system when fault/error handling is modeled. A discussion on this subject follows; however, the reader should see references 4 and 5 for more details.

When fault/error handling is considered, dependencies exist between stochastic events that make it necessary and practical to use a Markovian representation of the reliability model. A Markov process contains information about a system's fault processes, component depletion, and recovery procedures. Graphically, a Markov model consists of states and transitions. The states contain information about the number of operational components, and the transitions are rates at which specific components or subsystems fail causing a change in the state of the system. Computations are done to determine the probability of being in a state based on time. The reliability of the system can then be determined by adding the probabilities of the operational states (ref. 4). However, in systems designed with fault-tolerance, a very large state space model can result which introduces computational problems. These problems can be solved by utilizing the methods of decomposition and aggregation, i.e., dividing the system into smaller subsystems based on component types, solving these models separately, and then combining the results of the subsystem models to produce the larger system's solution. However, this method requires that the behaviors of the subsystems be independent. In many fault-tolerant systems this is a false assumption, because these systems may include dependencies. HARP, however, offers a unique modeling technique that surmounts this potential difficulty.

In addition to this traditional modeling technique, HARP offers a simpler and more efficient approach called behavioral decomposition (ref. 6). Using this method, HARP allows a user to segregate a reliability/availability model into two submodels, a fault-occurrence/repair model (FORM) and fault/error handling model (FEHM). The FORM describes a system as a fault tree or a Markov chain and relates information about hardware redundancy and fault processes. Using a FEHM to describe specific recovery procedures, a user can include details about permanent, transient, and intermittent faults in a reliability model. Figure 2 illustrates the behavioral decomposition method utilizing FORM and FEHM submodels. HARP provides a user with seven FEHMs which range from a simple probabilities and moments FEHM to a very complex extended stochastic Petri net FEHM. The model can be input into HARP by using an interactive textually oriented interface or a graphically oriented interface. If the FORM is a fault tree, it is first converted to a complex stochastic process that is reduced to a simpler Markov chain. The FEHMs are solved separately from the FORM to determine the exit probabilities and holding times for transient restoration, permanent coverage, near-coincident fault failures, and single-point failures. No matter how complex the FEHM models may be and no matter how many FEHMs are specified, this process will produce at most two additional system failure states in the chain which represent near-coincident fault failures and single-point failures. The reduction of an enormous number of Markov states for most practical systems is the forte of behavioral decomposition. The model is then given to a popular ordinary differential equation solver to compute the results.

HARP/Dynamic Fault Tree Modeling Engine

Input to HiRel takes one of two forms that can be either specified textually or graphically. In either case, the user can specify a FORM in the Markov graph or fault tree notation (ref. 7). The standard input to HiRel is the fault tree notation which consists of the standard combinatorial gates, AND, OR , and M out of N. Four special fault tree gates that allow sequence and functional dependencies have been added to provide a dynamic FORM modeling capability. The notational simplicity and power of these dynamic gates is demonstrated in references 8 and 9.

The functional dependency gate is depicted in figure 3. It is the logical equivalent of a combinatorial fault tree composed of AND and OR gates when no fault handling is specified. The input labeled "trigger" can be the output from any gate, whereas the outputs take two forms. The non-dependent output simply mimics the trigger input and may or may not be connected to any input of any gate, i.e., it can dangle if desired. The typical use of this gate involves the other outputs. The outputs labeled "dependent events" must be basic events. Although they are labeled dependent events, the basic events themselves are statistically independent. The dependency is related to the trigger event. A typical use of this gate is to account for the functional loss of devices because some other device failed and therefore is unable to provide signal or power input to the downstream operational devices.

A non-combinatorial gate that implements a cold spare model appears in figure 4. This sequence dependency gate is naturally called the cold spare (CSP) gate. The gate output fires (produces an output) when and only when the primary event occurs first followed by events 1st, 2nd, ..., nth. Events 1st, 2nd, ..., nth cannot occur first. Thus, the primary event can represent an active unit and event 1st is the cold spare that exhibits a zero failure rate until the active unit fails. At that instant, the cold spare is powered up and immediately exhibits a failure rate greater than zero. If additional cold spare units are added, they are powered up in the order of left to right and all inputs are independent basic events.

A useful variation of the CSP gate is called the sequence enforcing gate (fig. 5). The inputs of the sequence enforcing gate can be basic events or the output of some other gate for the primary input only. The sequencing of events is left to right similarly to the CSP gate. The cold spare gate and the sequence enforcing gate differ primarily in the way they treat shared events.

The last dynamic sequence dependency gate is called the priority AND (P-AND) gate (fig. 6). The P-AND gate differs from a combinatorial AND gate in only one respect: In HARP, only two inputs for the P-AND are allowed, and the gate produces an output only if the left most event occurs first followed by the right most event. Contrary to the CSP gate, the right most event in a P-AND can occur first, but no output results. The functionality, the name of the gate, and the gate symbol were obtained directly from the literature (ref. 10).

The developers experience with the use of these new gate additions to HARP has been extensive. They have applied them to some very complicated fault-tolerant network systems (ref. 8). Although there is no warm spare gate, that model has been functionally emulated with the existing gates, and pooled spares models have also been emulated. With HARP's Markov chain truncation technique that bounds the truncation error, extremely large Markov chains have been modeled and solved that have simple appearing fault tree diagrams. These models have demonstrated the succinct yet powerful notational value of HARP's dynamic fault tree capability.

An additional gate, the NOT gate, was added to HARP for completeness but was commented-out in the source code because its inclusion allows the modeling of noncoherent models. A noncoherent model allows the possibility of the top event of a fault tree to exhibit a decreasing probability of failure with increasing time. The HARP Team wanted to properly document the use of the NOT gate because of the likelihood of misuse. That documentation has not yet been completed; however, researchers at Duke University mathematically proved that the complete set of HARP's fault tree gates maps into the set of non-cyclic Markovian models with constant transition rates (ref. 9). Although there are no plans to further extend this capability at Langley, the most obvious and useful further extension should include a fault tree notation to model repair.

MC-HARP Modeling Engine

Simulation for use in reliability prediction has been used for decades. The traditional simulation method is called analog Monte Carlo which relies on a large number of failure events occurring in the mission time of interest. In highly reliable systems, the first system failure event occurring at time t is not likely to occur until $t > T$, where T is the mission time. Thus an inordinate number of simulation trials are required to produce an acceptable confidence level.

Recently, variance reduction techniques called importance sampling have been rediscovered by the reliability community. Importance sampling is a technique that was reported in the literature as early as 1984 (ref. 11). The basic concept of importance sampling is to force and bias transitions along the rare event paths in an underlying Markovian model (which may contain both local and global time dependence with a disparity of typically six orders of magnitude) while dynamically maintaining a record of the forcing and biasing that allows post simulation construction of an unbiased estimator of the event of interest, (e.g., system failure) with extremely low variance. The prime challenge over the years has been one of determining a suitable failure biasing scheme.

MC-HARP was developed to model non-Markovian models which arise when systems exhibit nonconstant failure rate histories and when cold or warm spares are employed (ref. 12). This failure history is a possible scenario for systems to be used in manned deep-space missions. The other motivation was to developed a modeling capability for correlated transient induced failures as might occur when an aircraft system is exposed to high intensity radio frequency emissions, e.g., lightning. Preliminary applications of MC-HARP as reported in reference 12 are very encouraging. Several highly reliable systems were analyzed and compared to the HARP analytical results. For large systems, MC-HARP proved more efficient particularly for non-constant failure rate models. MC-HARP can also be used to circumvent behavioral decomposition to serve as a check or to replace it.

PM-HARP Model Engine

Phase Mission HARP was developed by the University of Washington for Boeing Electronics and Aerospace Company (ref. 13). A phase is an epoch in a mission. During an epoch, a system may be altered by external means. An example of a phase occurs when the failure rates of the initial system change perhaps because of some environmental stress. A spacecraft system during launch would experience more vibration and shock than during orbital operation which would be a second and more benign phase. Another example of a phase mission occurs when a system is tested and repaired prior to the continuation of service of a commercial aircraft. During testing and repair, the system may not have been fully restored perhaps due to imperfect diagnostics or repair. The phase time may be deterministic or stochastic. PM-HARP was developed to facilitate this class of modeling and analysis.

XHARP Modeling Engine

Aside from the desirable portability of X-Windows HARP implemented in the X-Windows environment (X-HARP), X-HARP offers a unique automatic behavioral decomposition capability that was never implemented in the original HARP (ref. 14). The new fault/error handling modeling capability was developed to assist users who are unsure of the specifics of using the standard behavioral decomposition model. X-HARP further extends the multi-fault modeling capability of the original HARP to allow multiple entry and exit transitions to user specified fault/error handling models. Also, X-HARP allows the user to specify a detailed multi-fault model for system designs that use fault containment regions. Although the original HARP multi-fault models which were designed to be easy to use and specify will produce a conservative pessimistic unreliability prediction, for some system designs such as those with fault containment regions, the original HARP model may produce an overly conservative result. When an overly conservative result is unsatisfactory, X-HARP in conjunction with HARP, will produce a more accurate prediction commensurate with the accuracy of the user's data.

HIREL - Interactive Graphical User Interface

Graphics Oriented Interactive Input

The graphics oriented (GO) module enables the user to "draw" reliability models in the form of fault trees or Markov chains on the screen of a work station monitor (ref. 1). Figure 7 depicts the screen image for the fault tree drawing mode. A click of the "mouse" button device toggles the display to show the dynamic fault tree gates as shown in figure 8. A gate is drawn by selecting the draw primitive followed by selecting the particular gate to be drawn. Using the mouse, the cursor is positioned in the drawing screen area to the left, and the gate is moved to that position. Subsequent gates to be drawn are simply selected as required. The labeled squares on the right hand side of the screen in figures 7 and 8 are also selected with a cursor under control of a moving mouse device . The functions they provide are evident as labeled. Figure 9 displays the screen image for the Markov chain drawing mode. The drawing primitive provided by the four rectangles on the lower right are selected with a cursor as described previously to draw the chain model.

The captured image displayed on the screen serves as a documented copy of the model, as a hard copy can be readily made by the push of a button. The data from which the image was created also serves as the data for automatic input to the HARP reliability model solver. The introduction of dependency gates to a fault tree notation which provides a dynamic fault tree capability, allows the modeling of very large fault tolerant system models using a concise and visually recognizable and familiar graphical language. In addition to aiding in the validation of the reliability model, the concise graphical representation presents company management, regulatory agencies, and company customers a means of expressing a complex model that is readily understandable.

HARP Output

The graphical postprocessor module HARPO (HARP Output) makes it possible for reliability engineers to quickly analyze huge amounts of reliability/availability data to observe trends due to exploratory design changes (ref. 1). HARPO reads files automatically generated by the HARP modeling engine. It will accept files from any previously generated HARP executions for comparative analysis. The user can in an interactive mode display up to nine graphs representing modeling iterations of the same system or compare different system models. A number of parameters can be altered for analysis such as failure rate or coverage data for sensitivity analysis, or to view the effects of unreliability/unavailability as a function of different mission times. Markovian state probabilities or sums of user specified state probabilities can also be displayed. The user can manipulate HARP and HARPO ASCII files to do performability computations and display them. HARPO uses the ANSI standard Graphics Kernel System (GKS) graphics software which allows portability and provides a large number of device drivers to output graphical data to many hard copy devices, e.g., laser printers and plotters.

Figure 10 depicts a typical screen image for HARPO. The graph shows the probability of system failure versus time for two parameters of interest (M3F2 and M3REXHST) for the 5th version of model number 3 for a two processor - two bus system (MODEL3 5 3p2b). M3F2 designates "failure state number two" for model number three, where F2 is a failure state of the given Markov chain. M3REXHST designates the sum of all the failure states (including F2) that caused system failure resulting from the exhaustion of redundant hardware modules. In fault-tolerant systems, system failure can also be caused by improper fault/error handling not shown in this graph. The title RS - STATES tells the reader the data came from the RS (results) file generated by the HARPENG module.

HIREL PORTABILITY/AVAILABILITY

HARP was developed on a Sun 3 computing platform running under Berkley Unix 4.3. The source code was written in ANSI standard FORTRAN 77. HARP has been ported to a large host of computing platforms with the major operating systems being DEC VMS and Ultrix, Berkley Unix 4.3, AT&T Unix 5.2, and MS DOS. PC-HARP running under MS DOS is a scaled down version of HARP that executes on IBM compatible 286/386/486 machines. Certain limitations are placed on PC-HARP's capabilities because of the 640 K memory restriction imposed by MS DOS; however, extended FORTRAN compilers such as Lahey F77l-Em/16 and their DOS Extender are commercially available which use extended memory and thus breaches the 640K memory MS DOS barrier. Full scale HARP code can be compiled and executed without modeling restrictions on 286/386/486 PC compatible machines with extended memory. Operating systems other than MS DOS such as OS/2 and Unix, also allow full scale HARP to successfully execute with sufficient extended memory. Most of the HiRel software modules are available through NASA's software distribution facility, COSMIC[1] or from the developers at Duke University[2]. The MC-HARP[3], PM-HARP[4], and the XHARP[5] Engines are available from the universities where they were developed.

[1] COSMIC, The University of Georgia, 382 East Broad St., Athens, GA 30602 (404) 542-3265

[2] Duke University, Dept. of Computer Science, Durham, NC, 27706, Joanne B. Dugan, (919) 660-6559

[3] Northwestern University, Dept. Mechanical Eng., Evanston, IL 60206, E. E. lewis (708) 491-3579

[4] University of Washington, Dept. of E. E., Seattle, WA 98195, Arun. K. Somani (206) 685-1602

[5] Clemson University, Dept. of Computer Sci., Clemson, SC 29734-1906, Robert Geist (803) 656-2258

HIREL APPLICATIONS -

The core HiRel module, HARP, has been applied to numerous applications in the seven years of beta testing. Some of these applications are listed as follows:

Aircraft Life Critical Systems, Civilian Aircraft Electronics, Military Avionics, Space Systems, Computer Systems, Railroad Control Systems, Nuclear Power Control Systems, Submarine Steering Control Systems

For more detail on specific systems and architectures where HARP has been applied, see references 5, 7, 9, 10, 12, and 13. Also see the proceedings of the Duke/HARP Workshop (ref. 2).

REFERENCES

1. Salvatore J. Bavuso and Joanne Bechta Dugan: HiRel - Reliability/Availability Integrated Work Station Tool, Reliability and Maintainability Symposium Proceedings, Jan. 22 - 24 ,1992

2. K. S. Trivedi and S. J. Bavuso, compilers: HARP Workshop, Duke, University, July, 1990.

3. A. R. Timming: A Study of Total Space Life Performance of GSFC Spacecraft, NASA TN D-8017, 1975.

4. Joanne B. Dugan, Kishor S. Trivedi, Mark K. Smotherman, Robert M. Geist: The Hybrid Automated Reliability Predictor, Journal of Guidance, Control, and Dynamics, Vol. 9, No. 3, May-June 1986.

5. Salvatore J. Bavuso, Joanne B. Dugan, Kishor S. Trivedi, Elizabeth M. Rothmann, and Mark A. Boyd: Applications of the Hybrid Automated Reliability Predictor, NASA TP 2760, December 1988.

6. Robert Geist, Mark Smotherman, Kishor Trivedi, and Joanne Bechta Dugan: The Reliability of Life Critical Systems, Acta Informatica 23, 621-642 (1986).

7. Elizabeth M. Rothmann, Joanne Bechta Dugan, Kishor S. Trivedi, Mark A. Boyd, Nitin Mittal, Salvatore J. Bavuso: HARP: The Hybrid Automated Reliability Predictor Introduction and Guide for Users, Version 6.1, Dept. of Computer Science, Duke University, May 1990.

8. Joanne Bechta Dugan, S. J. Bavuso, and M. A. Boyd: Modeling Advanced Fault-Tolerant Systems with HARP, Tutorial Presented at the Annual Reliability and Maintainability Symposium, January, 1991.

9. Mark A. Boyd: Dynamic Fault Tree Models: Techniques for Analysis of Advanced Fault Tolerant Computer Systems, Phd Dissertation, Dept. of Computer Science , Duke University, April, 1991.

10. J. B. Fussell, E. F. Aber, and R. G. Rahl: On the Quantitative Analysis of the Priority-AND Failure Logic, IEEE Transactions of Reliability, R-25(5), 324-326, December 1976.

11. E. Lewis and F. Bohem: Monte Carlo Simulation of Markov Unreliability Models, Nuclear Engineering and Design, 77 (1984).

12. M.E. Platt, E. E. Lewis, and F. Bohem: General Monte Carlo Reliability Simulation Code Including Common Mode Failures and HARP Fault/Error Handling, Dept. of Mechanical Engineering, Northwestern, University, Final Report for NASA Grant NAG-1-1031, January 1991.

13. Arun K. Somani, James A. Ritcey, Stephen H.L. Au, and Hamid Amindavar: Phased Mission Analysis Program Users Manual, Dept. of Electrical Engineering, University of Washington, December 1989.

14. Robert Geist: Extended Behavioral Decomposition for Estimating Ultrahigh Reliability, IEEE Transactions on Reliability, Vol. 40, NO. 1, April, 1991.

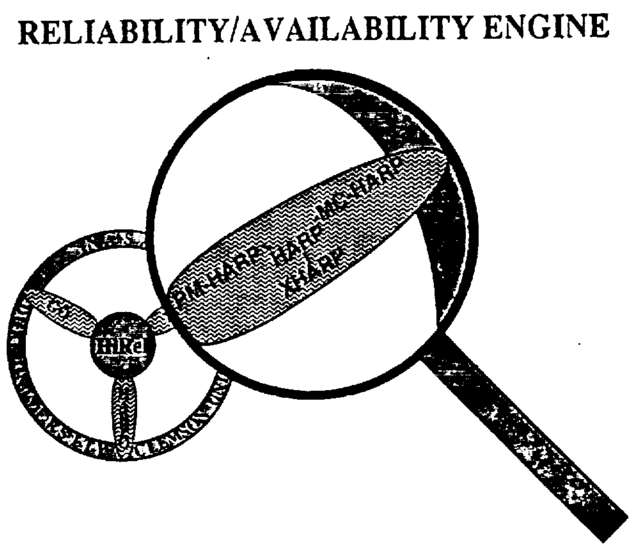

Figure 1: HiRel tool system depicting the modeling engine suite

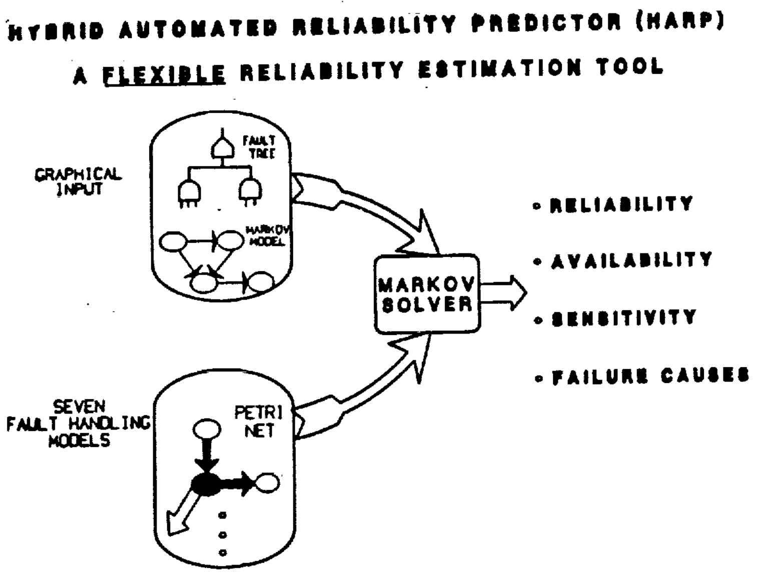

Figure 2: HARP/behavioral decomposition

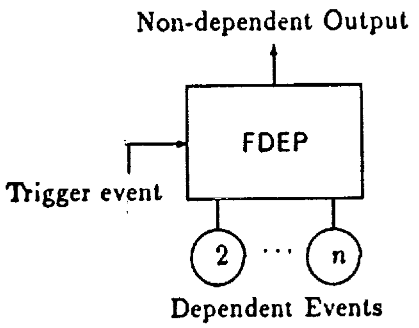

Figure 3: Functional dependency gate

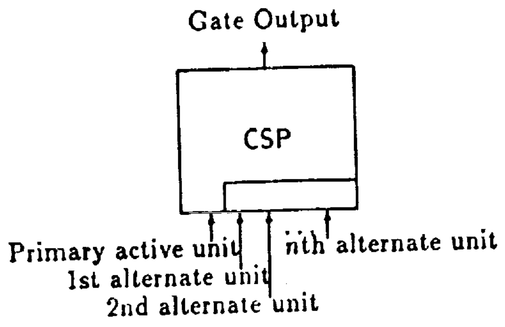

Figure 4: Cold spare gate

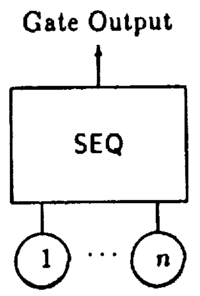

Figure 5: Sequence enforcing gate

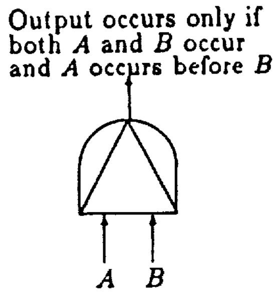
Output occurs only if both A and B occur and A occurs before B
A
B

Figure 6: Priority-AND gate

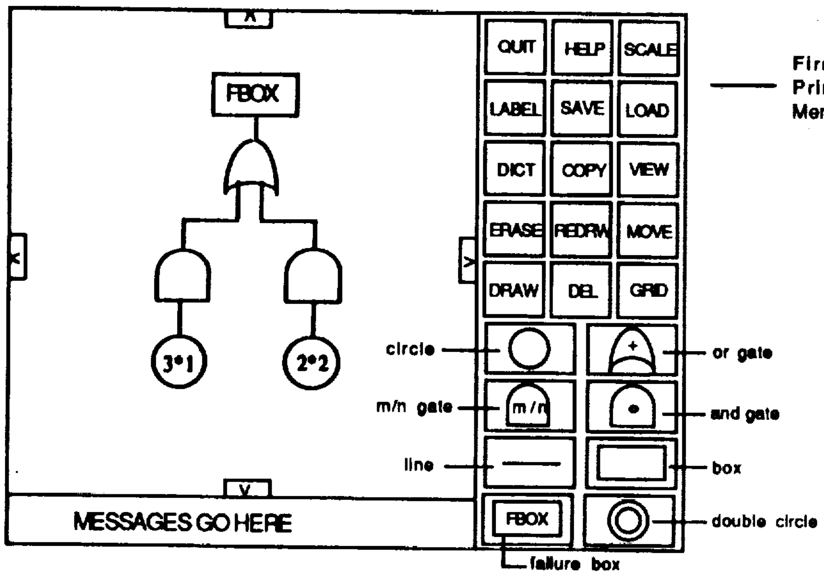

Figure 7: GO Module Screen Image for a Fault tree model

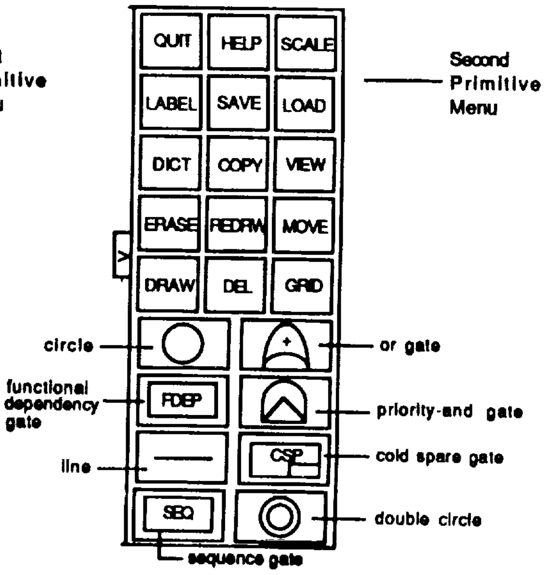

Figure 8. Primitive Menu for Sequence Dependency Gates

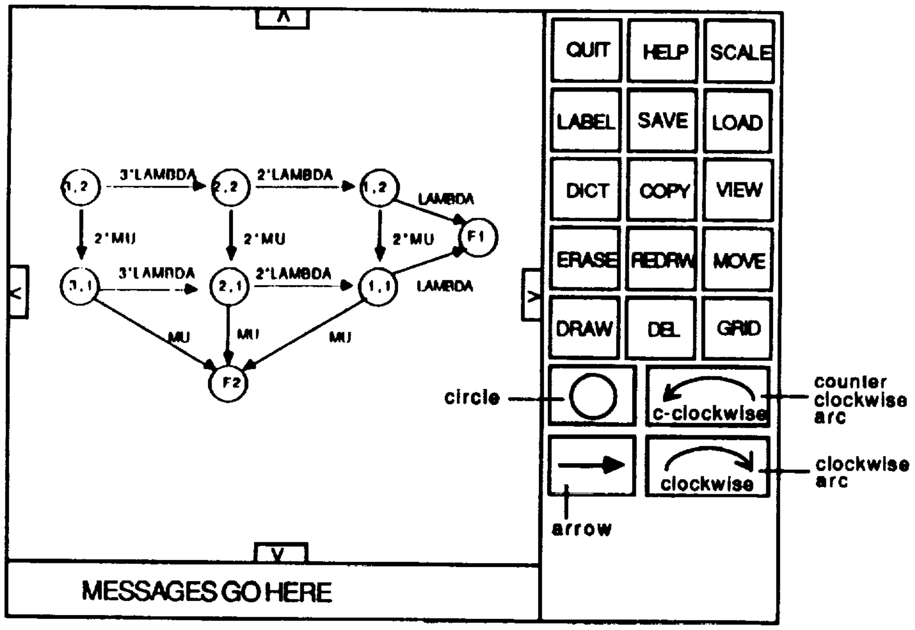

Figure 9. GO Module Screen Image for a Markov Chain Model

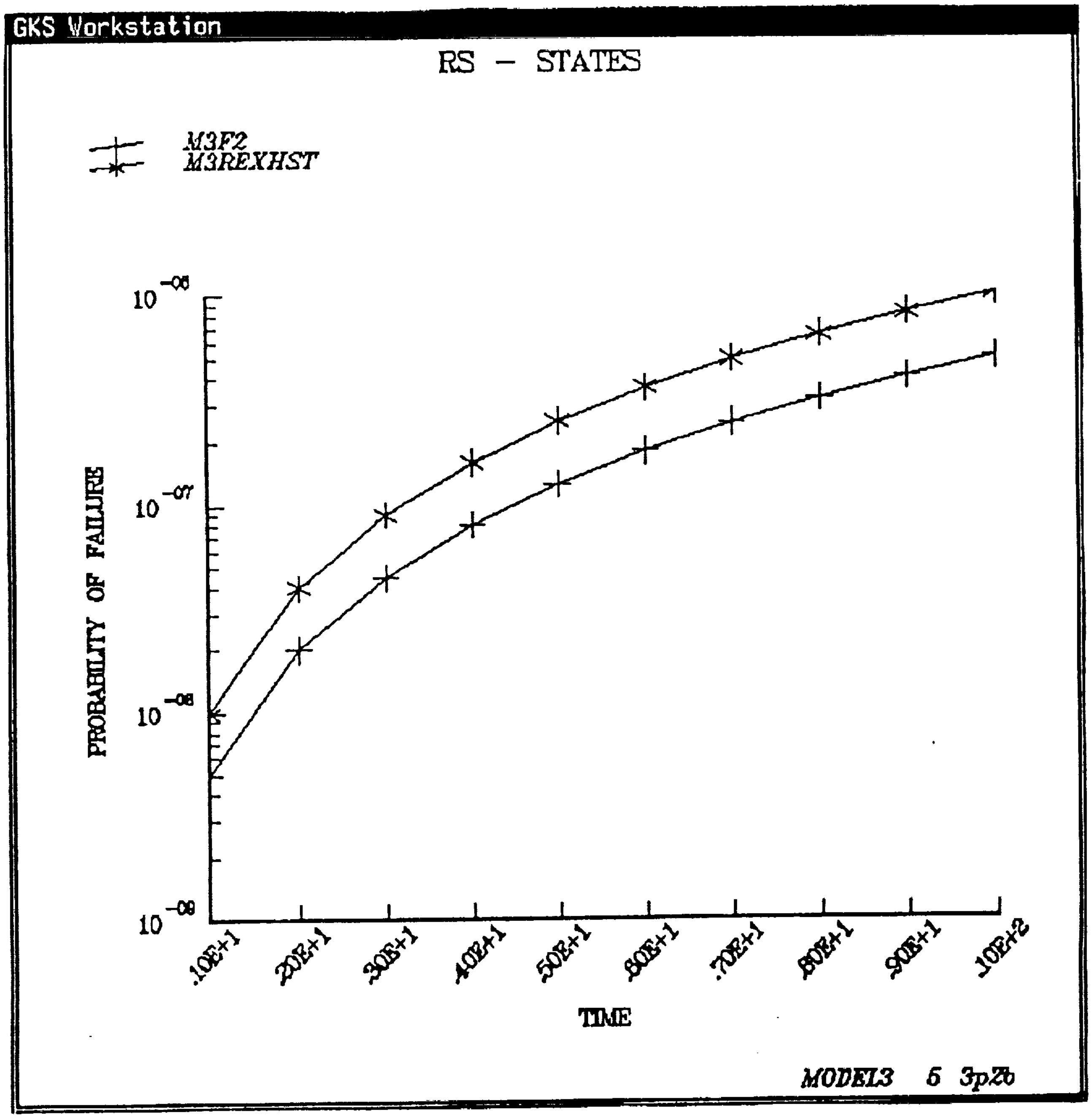

Figure 10. HARPO Module Screen Image for a 3-processor, 2-Bus System