UNIVERSITY OF CALIFORNIA

Los Angeles

Integration of Qualitative and Quantitative Hybrid Causal Logic into a Simulation-based Platform for Probabilistic Risk Assessment of Nuclear Power Plants

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Mechanical Engineering

by

Mihai Aurelian Diaconeasa

2017

ABSTRACT OF THE DISSERTATION

Integration of Qualitative and Quantitative Hybrid Causal Logic

into a Simulation-based Platform

for Probabilistic Risk Assessment of Nuclear Power Plants

by

Mihai Aurelian Diaconeasa

Doctor of Philosophy in Mechanical Engineering

University of California, Los Angeles, 2017

Professor Ali Mosleh, Chair

Dynamic Probabilistic Risk Assessment (PRA) refers to an emerging class of PRA methods that generate risk scenarios through the model-based simulation of systems such as nuclear power plants (NPPs) and their crew response to accident initiators. The dynamic PRA approach offers several advantages over the conventional approaches currently used by the nuclear industry worldwide. These advantages include: (1) time-dependent prediction of the operator error-forcing contexts, (2) better representation of the thermal-hydraulic success criteria, and (3) considerable reduction in analyst-to-analyst variability of the results. An example of such a simulation platform is the Accident Dynamics Simulator coupled with the Information, Decision and Action in a Crew context cognitive model (ADS-IDAC), and a realistic NPP thermal-hydraulic model. The aim of this research is to integrate qualitative and quantitative hybrid causal logic into the ADS-IDAC dynamic PRA platform. This makes ADS-IDAC a more practical and realistic analysis tool for specific applications. These applications are primarily event assessments, but also include the ability to analyze highly dynamic and complex accident scenarios in support of conventional PRAs. This work offers major modeling enhancements of ADS-IDAC, including dynamically linked fault trees (FTs) for support and frontline systems modeling, more advanced system and operating crew modeling capabilities, comprehensive quantification features modeling human failure evens (HFEs), and uncertainty propagation through the generated discrete dynamic event tree (DDET). The new risk assessment process was streamlined with the help of a newly developed user-friendly graphical interface, which provides efficient and convenient access to all the capabilities of the ADS-IDAC simulation engine.

The dissertation of Mihai Aurelian Diaconeasa is approved.

B. John Garrick

Ertugrul Taciroglu

Ivan Catton

Ali Mosleh, Committee Chair

University of California, Los Angeles

2017

Dedication

This work would have been impossible without the patience and support of my loving wife, Agatha Diaconeasa. I am also grateful to the motivation, work ethic, and the learning as lifetime mission inspired by my parents, family, and educators from an early age.

This dedication would not be whole without mentioning the contributions to my career path that came from mentoring and friendship from Dr. Vesna Dimitrijevic and Prof. Michael Golay of Massachusetts Institute of Technology (MIT). I would also like to recognize Prof. Ali Mosleh and Prof. B. John Garrick of University of California Los Angeles (UCLA) for their friendship, guidance, and inspiration to not only relentlessly pursue this research, but also helping me understand its historical context and importance.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADAPT	Analysis of Dynamic Accident Progression Trees
ADS	Accident Dynamics Simulator
AFW	Auxiliary Feedwater System
ASP	Accident Sequence Precursor
AT	Action Taker
BBN	Bayesian Belief Network
CCDP	Conditional Core Damage Probability
CCW	Component Cooling Water
CET	Continuous Event Tree
CFR	Code of Federal Regulations
COSIMO	Cognitive Simulation Model
CREAM	Cognitive Reliability Error Analysis Method
CVC	Chemical and Volume Control
DBN	Dynamic Bayesian Network
DDET	Discrete Dynamic Event Tree
DETAM	Dynamic Event Tree Analysis Method
DM	Decision Maker
ECCS	Emergency Core Cooling System
EDG	Emergency Diesel Generator
EOC	Error of Commission
EOP	Emergency Operating Procedure
ET	Event Tree
ESD	Event Sequence Diagram
FCV	Flow Control Valve
FRG	Functional Recovery Guideline
FT	Fault Tree
HAMMLAB	Halden Human-Machine Laboratory
HCL	Hybrid Causal Logic
HEP	Human Error Probability
HFE	Human Failure Event
HRA	Human Reliability Analysis
IAEA	International Atomic Energy Agency
IDAC	Information, Decision, and Action within a Crew cognitive context
LOCA	Loss of (Reactor) Coolant Accident
LOFW	Loss of Feedwater
LOSC	Loss of Seal Cooling
LWR	Light Water Reactor
MCS	Monte Carlo Sampling

MDAFP	Motor Driven Auxiliary Feedwater Pump
MSIV	Main Steam Isolation Valve
MSLB	Main Steam Line Break
NPP	Nuclear Power Plant
NRC	Nuclear Regulatory Commission
PSF	Performance Shaping Factor
PORV	Power Operated Relief Valve
PRA	Probabilistic Risk Assessment
PWR	Pressurized Water Reactor
PZR	Pressurizer
RAVEN	Risk Analysis Virtual Environment
RCP	Reactor Coolant Pump
RCS	Reactor Coolant System
RHR	Residual Heat Removal
RO	Reactor Operator
RPD	Recognition Primed Decision-Making
RWST	Refueling Water Storage Tank
SACADA	Scenario Authoring, Characterization, and Debriefing Application
SGTR	Steam Generator Tube Rupture
SI	Safety Injection
SRO	Senior Reactor Operator
SUT	Startup Transformer
TDAFP	Turbine Driven Auxiliary Feedwater Pump
THERP	Technique for Human Error Rate Prediction
TMI	Three Mile Island Nuclear Power Station
UAT	Unit Auxiliary Transformer
VCT	Volume Control Tank

Acknowledgements

This work was funded through a Research Grant (NRC Grant HQ-60-14-G-0013) by the United States Nuclear Regulatory Commission (NRC) and by the NASA's Jet Propulsion Laboratory (JPL).

I would also like to acknowledge the technical advice and encouragement from the following people and organizations: UCLA's Prof. Ali Mosleh, Prof. B. John Garrick, Prof. Ivan Catton, Dr. Christopher Jackson, Dr. Elaheh Rabiei, NRC's Ms. Michelle M. Gonzalez, Dr. Kevin Coyne, Mr. Christopher Hunter, Mr. Donald Helton, Mr. Christopher Murray, and Mr. Douglas Barber, JPL's Ms. Melissa Meyers, Dr. Mark White, and Dr. Harald Schone, and MIT's Prof. Koroush Shirvan and Prof. George Apostolakis.

Vita

Education

2014	M.S. Nuclear Science and Engineering Massachusetts Institute of Technology Cambridge, MA
2010	B.S. Mathematics, Physics, Chemistry University College Utrecht Utrecht, the Netherlands

Research Experience

Employer: Nuclear Science and Engineering, Massachusetts Institute of Technology, USA

Job function: Research Assistant: "An information theory based approach for human reliability improvement in complex systems"

Start Date: 2013/05; *End Date:* 2014/08

Supervisor's Name: Prof. Michael Golay

Employer: National Research Nuclear University "MEPhI", Moscow, Russia

Job function: Visiting Research Assistant: "A fuzzy theory based approach for human reliability improvement in complex systems"

Start Date: 2013/06; *End Date:* 2013/08

Supervisor's Name: Prof. Vladimir Kosterev and Prof. Michael Golay

Employer: Nuclear Science and Engineering, Massachusetts Institute of Technology, USA

Job function: Research Assistant, Master's Thesis: "Development and optimization of the MIT Reactor LEU fuel element design using Computational Fluid Dynamics"

Start Date: 2012/02; *End Date:* 2013/05

Supervisor's Name: Prof. Emilio Baglietto

Employer: Nuclear Science and Engineering, Massachusetts Institute of Technology, USA

Job function: Undergraduate Researcher: "Modeling of hydrogen diffusion and hydride evolution in Zircaloy-4" part of the CASL Project

Start Date: 2010/09; *End Date:* 2011/05

Supervisor's Name: Prof. Mujid Kazimi

Employer: Van't Hoff Laboratory for Physical and Colloid Chemistry, Debye Institute for Nanomaterials Science, University Utrecht, the Netherlands

Job function: Undergraduate Researcher, Honors Thesis: "Subtle effects of goethite nanoparticles in a magnetic field"

Start Date: 2010/01; *End Date:* 2010/05

Supervisor's Name: Prof. Gert Jan Vroege

Employer: Institut für Photovoltaik (IEF-5), Forschungszentrum Jülich, Germany
Job function: Undergraduate Researcher: "Flexible thin-film silicon solar cells in the Helianthos Project – KeyHole Project"

Start Date: 2009/06; *End Date:* 2009/09

Supervisor's Name: Dr. Ümit Dagkaldiran

Employer: Robert van de Graaff Laboratory, Debye Institute for Nanomaterials Science, University Utrecht, the Netherlands

Job function: Undergraduate Researcher: "Magnetron sputtered and texture etched aluminum-doped zinc oxide films for silicon thin film solar cells on plastic"

Start Date: 2008/12; *End Date***:** 2009/04

Supervisor's Name: Prof. Jatin Rath

Teaching Experience

2015-2017	Grader and Guest Lecturer MSE.298 "Risk Analysis for Engineers and Scientists" given by Prof. Ali Mosleh at UCLA Materials Science and Engineering
2015-2017	Grader and Guest Lecturer MAE.174 "Probability and Its Applications to Risk, Reliability, and Quality Control" given by Prof. Ali Mosleh at UCLA Mechanical and Aerospace Engineering
2012	Grader and Guest Lecturer for 22.38 "Probability and Its Applications To Reliability, Quality Control, and Risk Assessment" given by Prof. Michael Golay at MIT Nuclear Science and Engineering
2006, 2007	Member of the Examination Commission, National Mathematics Contest "Nicolae Coculescu" (high school level), Romania
2004, 2005	Member of the Examination Commission, National Mathematics Contest "Nicolae Coculescu" (middle school level), Romania
2004, 2005	Member of the Examination Commission, National Mathematics Contest "Matematica – Olt" (middle school level), Romania

1 Introduction

1.1 Motivation

Nuclear power plants (NPPs) present several unique hazards to the health and safety of the public. Foremost, the nuclear reactor core contains radioactive fission products that, if released, pose a serious threat to public health and the environment. Even after a reactor core shutdown, fission product radioactive decay continues to produce a substantial amount of heat that must be removed to prevent core damage and subsequent radiological release. Although NPPs have automated systems whose goal is to prevent fission product release and provide core cooling, the NPP operators still play a vital role in NPP safety.

The current International Atomic Energy Agency (IAEA) safety standards for protecting people and the environment mandate assessments that determine the NPP reliance on safety functions for normal operations, accident conditions, or beyond design basis accidents. Both deterministic and probabilistic analysis methods are applied to assess the challenges to the safety of the design in question (i.e. the defense in depth philosophy).

A deterministic safety analysis is defined as the analytical evaluation of the physical phenomena that typically occur in response to a bounding set of fault conditions or a postulated initiating event. Depending on the computational tools used, these analyses focus on neutronic, radiological, thermo-hydraulic, thermo-mechanical or structural phenomena. A strict set of rules and acceptance criteria are applied for each type of phenomenon, with uncertainty addressed via conservative assumptions. Deterministic safety analyses help provide or validate strategies to restore normal operational conditions following transients or design basis accidents. These strategies are tabulated in the emergency operating procedures (EOPs) that aid the operators in making diagnoses, decisions, and taking the appropriate actions to prevent further risk. Theoretically, the safety analyses should be able to identify all the relevant physical phenomena and the potential challenges that the operators would encounter during the progression of accidents under the postulated conditions. This approach helps confirm if acceptance criteria are met, but can produce misleading results that may not be appropriate for evaluating a realistic plant risk, especially for events with a low frequency of occurrence. Therefore, probabilistic analysis methods have been developed to cover a wider range of fault conditions, benefit from the deterministic analyses tools, and capture inter-dependencies between the systems, thus providing more realistic accident scenarios to facilitate better risk-informed decisions and increase the design safety.

In the application of the defense in depth concept, PRA helps determine the probability for breaching each barrier. PRA methods are generally employed to identify failure scenarios and to estimate their associated risk by answering to three fundamental questions (Kaplan and Garrick 1981): 1) “What can happen?”, 2) “How likely it is that it will happen?”, and 3) “If it does happen, what are the consequences?” System reliability analysis, human reliability analysis, and uncertainty analysis methods are explicitly integrated into the PRA framework, thereby enabling the application of defense in depth, which would be impossible to do using deterministic analysis methods alone.

The conventional approach to perform a PRA is to combine event trees (ETs) and fault trees (FTs). An event tree (Figure 1-1) is an inductive logic method that captures the high-level characteristics of accident sequences that arise from an initiating event and, given the failure or success of the main safety systems or operator actions that are also called ‘top events’, predicts NPP sequence end state probabilities. The end states could be a safe outcome or one of the plant damage states. An FT is a logical failure model that is developed to evaluate the success or failure state of a top event in the ET (Figure 1-1). In graph theory, an FT is an acyclic graph with logic gates and basic events that represent system events. FTs typically include the relevant individual components and operator errors that could lead to the failure of the top event, which is generally some component or subsystem failure, or human event. The size of the ET and the depth of the FT are subjectively set by the analyst depending on the available resources and the end goal.

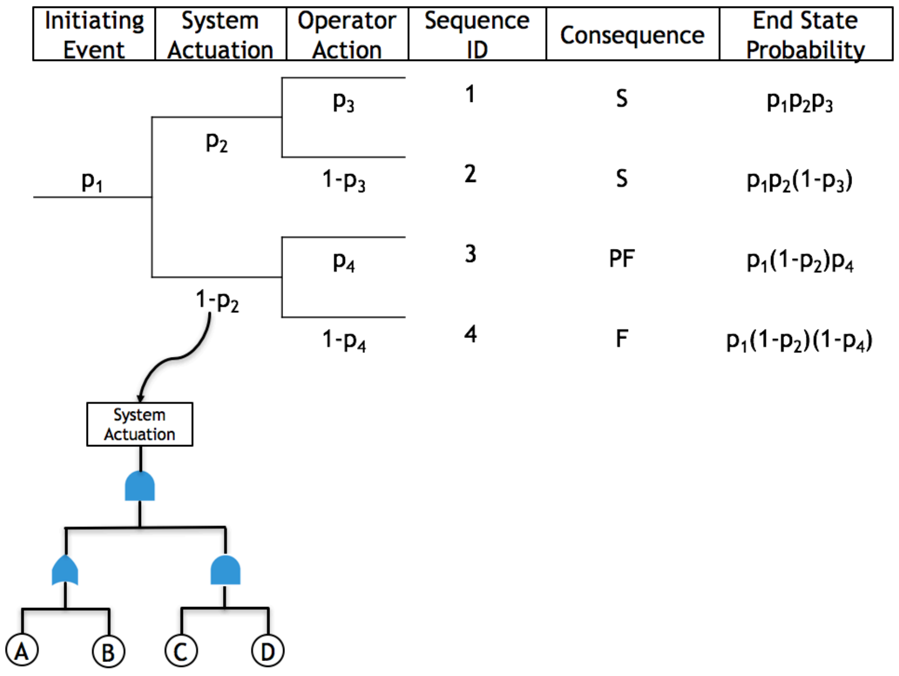


Figure 1-1 Example of an ET with two top events and an FT with its four basic events

In parallel with the development of ETs and FTs, human reliability analysis (HRA) methods emerged as a necessary tool to estimate the failure probabilities of human operators (Swain, 1983). First-generation HRA methods assumed that human behavior can be decomposed like a hardware system. These models were very dependent on the analyst who would have to identify the appropriate human failure modes and assign their probabilities based on the context by evaluating a set of performance shaping factors (PSFs). One such commonly used method is the Technique for Human Error Rate Prediction (THERP). THERP is a straight-forward method that offers the necessary guidance to perform the analysis, including a wide range of error modes and their suggested probabilities. However, THERP does not account for the time-dependency of human performance, nor the exploration of the cognitive mechanism behind the human behavior. The latter shortcoming was only partially addressed in the second-generation HRA methods. The Cognitive Reliability Error Analysis Method (CREAM) is based on a classification scheme that can be used to find the likely causes for a given human event by mapping the human errors and the various cognitive processes they tie into (Hollnagel, 1998).

The first full scale application of PRA to a large complex system was the Reactor Safety Study WASH-1400 (1975). Nuclear regulators and licensees have since performed PRAs on more than 200 NPPs worldwide. Historically, PRA results provided a legal basis to resolve regulatory issues. For example, full scope PRAs were performed for the Zion (1981) and Indian Point (1982) NPPs to determine the safety adequacy of their design after a petition by opponents to the plants claimed costly retrofits should be installed to reduce the risk of severe accidents (Garrick, 2013). The results of those PRAs showed not only that the claimed retrofits had a negligible influence on risk, but also several low-cost changes were transparently identified to actually reduce the overall risk. PRAs have also been used in the aviation and aerospace industries to facilitate decision making.

Certain concerns linger regarding the capability of conventional PRA methods to completely represent the interactions between NPP physical processes, hardware systems, software, and operators (Siu, 1994). Additionally, the order of the top events was shown to depend not only on the causal relationship between them, but also on their timing and the precise magnitude of the process variables at the time of the failure events. Time independent ETs and FTs are limited in providing contextual information for the HRA tools. Furthermore, a PRA study heavily relies on the expertise of the analyst, which can sometimes result in an incomplete set of risk significant accident scenarios. To overcome these challenges and shortcomings, various hybrid or dynamic

PRA methodologies, also known as 'simulation-based' PRA methodologies, have been proposed (Mosleh, 2014).

Dynamic PRA methodologies are generally those that use a time-dependent phenomenological model of system evolution and consider stochastic behavior to estimate the risk associated with the system response to an initiating event (Aldemir, 2013). The system evolution model keeps track of the current hardware status, current level of processes variables, current operator assessment, scenario history, and time (Siu, 1990). A graphical representation of the system evolution space with its probabilities of occurrence is shown in Figure 1-2.

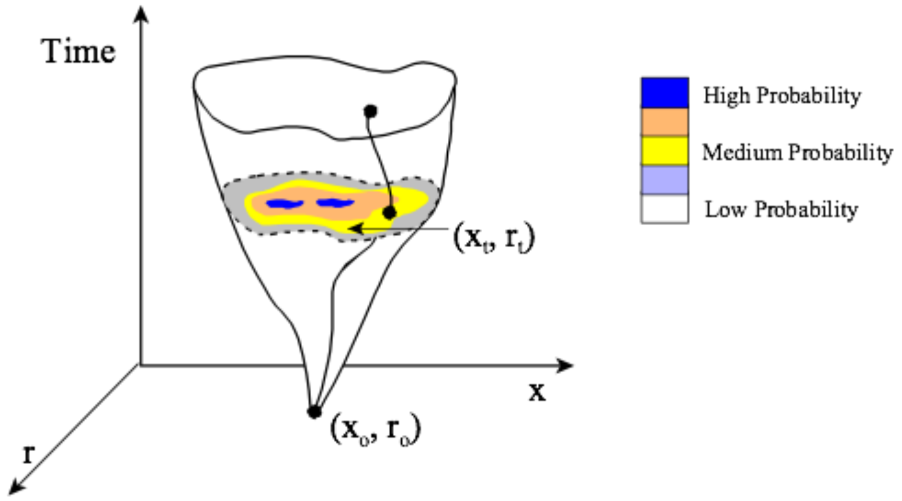


Figure 1-2 Time dependent system evolution under uncertainty (Mosleh, 2015)

Dynamic PRA has been grouped in two main categories: continuous-time (e.g. Continuous Event Tree (CET)) and discrete-time (e.g. Dynamic Event Tree Analysis Method (DETAM), Accident Dynamic Simulator (ADS), Analysis of Dynamic Accident Progression Trees (ADAPT), and Risk Analysis Virtual Environment (RAVEN)). The CET models simulate the possible dependencies between the process variables, hardware, software, and human interactions with a single integral equation generally solved with Monte Carlo techniques (Devooght and Smidts, 1996). Most of the discrete dynamic PRA methodologies use discrete dynamic event trees (DDETs) that are computationally generated based on a time-dependent model of system evolution and various branching conditions. Essentially, all discrete dynamic PRA methodologies employ a simulation engine that generates branches at each user-specified time step or conditions with their associated probabilities and computes the probability of each scenario. As can be seen in Figure 1-3, branching points can include system hardware states, physical variable changes, human actions, software failures or an end state if one of the stopping criteria is met. Hence, the resulting tree is called DDET.

The dynamic methodologies provide a natural framework to include physical models, mechanistic models of hardware failure or operator behavior models. One such methodology for NPPs is the Accident Dynamics Simulator coupled with the Information, Decision, and Action in a Crew context model (ADS-IDAC).

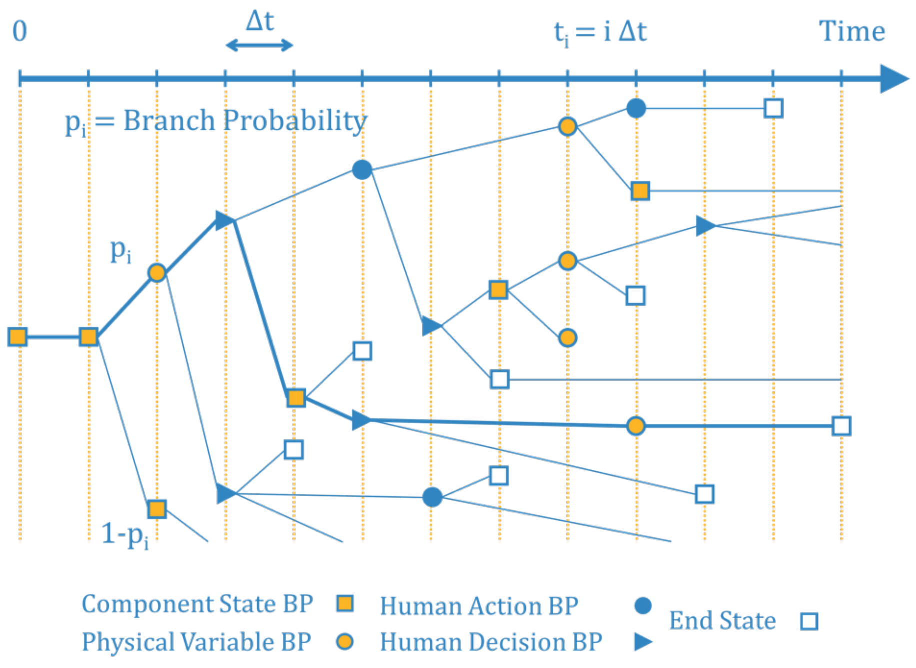


Figure 1-3 Discrete dynamic PRA methodology

Although the ADS-IDAC's feasibility has been successfully ascertained, it still lacks certain features to be successfully used for more realistic applications and break the cultural barrier of moving from conventional PRA towards simulation-based PRA. This research attempts to resolve these challenges.

Dynamic PRA has its own limitations too, amongst them the resource intensive and difficult validation of physical models, the assurance of a complete solution space, the aggregation and identification of the most risk significant scenarios, and dynamic uncertainty analysis methods. From a regulatory perspective, the integration, identification of appropriate figures of merit, and communication of simulation-based PRA results into the decision-making context are some of the challenges recognized by (Coyne and Siu, 2013).

Despite these challenges (Coyne and Siu, 2013) still see some near term regulatory opportunities for dynamic PRA: event and condition assessment for scenarios that involve complex dependencies, degraded equipment, and variability in human response, support expert judgment based decision-making, and provide insights to augment conventional PRA modeling.

The research gaps that are explored and filled in this thesis are identified and highlighted in the following subsections.

1.1.1 Impact of support systems on the frontline systems

The ADS-IDAC has limited applicability for scenarios where the support systems have a great impact on the frontline systems. The support systems are currently unsupported both in the system model and the hardware reliability module. This can be achieved by dynamically linking FTs for the support systems to the hardware failure branching points in the DDET. A static implementation of this approach was successfully done in the Hybrid Causal Logic (HCL) methodology for event sequence diagrams (ESDs), a variant of a classical ETs, which was initially developed to extend conventional risk analysis models that were limited to the physical systems and physical environment by including factors related to human events, socio-economical, or regulatory environment (Wang, 2007). The dynamic linking of the FT for the time-dependent states of the support systems can be achieved by developing an algorithm for incorporating the binary logic of system failures into the dynamic branching rules of the DDET.

1.1.2 Hybrid procedure and knowledge-based model of operator behavior

The IDAC cognitive model is a simulation-based HRA method with a causal model of cognitive processes that utilizes the advances in cognitive psychology, behavioral sciences, neuroscience, human factors, field observations, and previous experience in the development of the first and second-generation HRA methodologies. As its name suggests, the 'Information, Decision, and Action in a Crew context cognitive model' considers the time-dependent operator behavior in a holistic response to the system abnormal conditions. NPP operators work in a highly regulated environment with extensive training, procedures or other boundary conditions, thus greatly reducing the complexity of the problem of modelling cognitive human behavior.

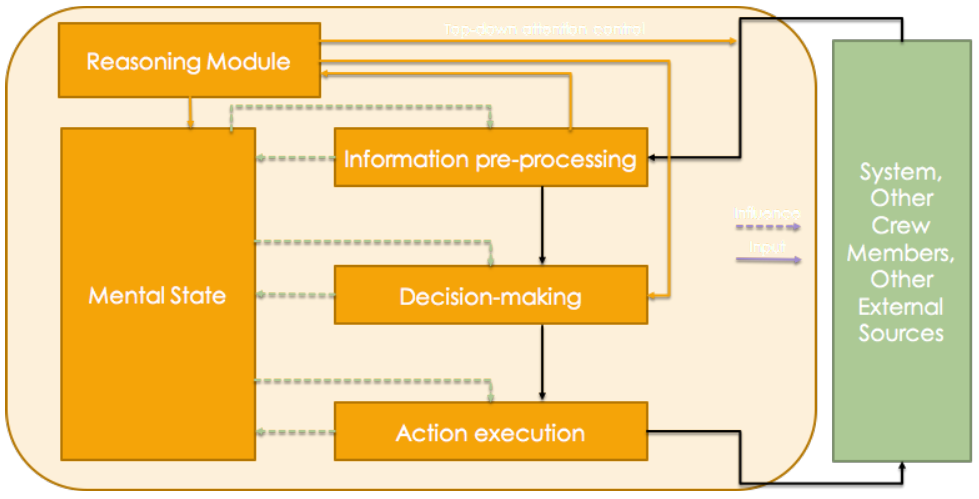

Reasoning Module
Top-down attention control
Mental State
Information pre-processing
Decision-making
Action execution
Influence
Input
System, Other Crew Members, Other External Sources

Figure 1-4 IDAC Modules and information exchange paths

Many important features have been added or improved to augment the predictive power and realism of the IDAC model. The current state of the IDAC model is shown in Figure 1-4. The main modules show the operator's dynamic loop of response phases at each time step: information pre-processing, decision-making, and action execution. The information pre-processing module filters the incoming information to simulate the limitation of human perception. Through the 'Mental State' as the engine of cognition, the context and the information gathered are used to generate a probabilistic response of the operator through explicit causal chains. The context and operator variability are implemented through either static or dynamic PSFs (Table 1-1).

Table 1-1 PSFs included in the IDAC model (Li, 2013)

PSF	Quantitative Assessment	Mechanism Modeling	Dynamic/Static	Code Version
Attention		X	D	2.0/3.0*
Problem-solving style		X	D	3.0*
Prioritization		X	D	3.0*
Information use		X	D	2.0/3.0*
Time load	X		D	2.0/3.0*
Task load	X		D	3.0*
Expertise (Knowledge/experience/training)	X	X	S	3.0*
Passive information load	X		D	3.0*
Information load	X		D	2.0/3.0
System criticality	X		D	2.0/3.0
Task complexity	X		D	3.0*
Stress	X		D	3.0*
Fatigue	X		D	3.0*

IDAC also includes an advanced reasoning module and structured knowledge representation for replicating the operator behavior governed by both their knowledge of the NPP gathered through training or experience and the EOPs. The reasoning module uses a top-down operator attention mechanism to actively gather information required in knowledge-based decision-making. Based on the perceived NPP state and the situational assessment, the decision-making module generates high level goals to guide further activities. The operator attempts to achieve these goals by implementing an appropriate problem-solving strategy. Finally, in the action execution phase, the necessary actions are taken according to the problem-solving strategy.

Previous research efforts have found that NPP operators can be induced to commit unsafe actions under certain error forcing situational contexts. Situational context includes the system state, the operator’s state of mind, and the sequence and timing of events.

A simplified version of the hybrid response mode will be developed and implemented to replicate a knowledge-based procedure-following response mode for improving the realism of how to operators behave in abnormal conditions especially for contexts that lead to human error of commission. These are situations when the operators took the appropriate decisions and actions given their understanding of the NPP conditions in a highly uncertain environment, but were not the best measures that could have been taken given the real NPP conditions.

1.1.3 Full dynamic ET quantification

In most of the HRA methods, the human error probability (HEP), defined as the probability of an operator not completing a specific task, is quantified as a function of the PSFs. In ADS-IDAC, the PSFs are quantified in terms of their contextual parameters (i.e. surrogates) and their impact on the cognitive processes is implemented through manifestation nodes (Li, 2013).

The PSFs are not assumed to be independent. Although it greatly improved the explicit impact of the PSFs on the human performance, ADS-IDAC still lacks a full implementation for explicitly quantifying the HEPs based on the dynamic nature of the PSFs. For each individual or team activity, the behavioral effects of the PSFs can be accounted for through an influence diagram. Like its application in the Phoenix method (Ekanem, 2013), the Bayesian belief network (BBN) approach can be used to estimate the probability that a specific cognitive behavior occurs given certain conditions.

1.1.4 Uncertainty quantification

The two major types of uncertainties are aleatory and epistemic uncertainties. Aleatory uncertainties account for the randomness in the behavior of a system or crew, while epistemic uncertainties arise from a lack of knowledge of the systems, processes or mechanisms. Dealing with the aleatory uncertainties is relatively straightforward when data is available and allows probabilistic characterization. Probability-based approaches such as Monte Carlo simulations are typically used to describe uncertainties in input data and propagate them through FTs or ETs. This technique requires empirical input data information or expert judgement in the form of probability density functions of relevant parameters. In some cases, this information may not be available. As an alternative to empirical data that can be difficult to obtain, expert judgment is typically used. Monte Carlo simulations can be computationally expensive, requiring efficient sampling techniques.

1.1.5 Graphical user interface

The ADS-IDAC engine typically requires a large amount of information and generates a large number of events. The ability of users to easily input the necessary data, post-process, aggregate, and visualize the results is of paramount importance for discrete dynamic PRA simulation platforms. Graphical software enhancements to support the previous and current tasks are amongst the necessary features needed to be implemented into the ADS-IDAC graphical user interface.

1.1.6 Accident Sequence Precursor (ASP) Analysis

The ASP Program is one of the critical United States Nuclear Regulatory Commission (NRC) programs focused on continuously assessing the risk significance of performance deficiency or degraded conditions. It mainly uses retrospective event and condition assessment to identify and rank the operational events that could potentially lead to a core damage state and eventually release of fission products. Conventional PRA employing the ETs – FTs remains the standard approach used to support these activities. There is no simulation-based PRA platform developed that can facilitate ASP. An ASP is defined as an observed event that combined with one or several postulated events could lead to core damage. An ASP analysis is a PRA performed to obtain the conditional core damage probability (CCDP) given an initiating event and identify the dominant event sequences (Cheok, 2004). The use of simulation-based PRA tools such as ADS-IDAC can aid in solving some of the challenges of conventional methodologies. However, it is critical to emphasize that the reasons for doing dynamic ASP, that is ASP analysis with dynamic PRA tools are mutually beneficial. The narrow scope of an ASP analysis aligns particularly well with the dynamic PRAs that can be resource intensive since the analysis is intended to be limited only to the operational event of interest with a few postulated complications. Therefore, one of the drivers of this work is to enhance ADS-IDAC and show how it can be used for performing dynamic ASP.

1.2 Objectives of the research

The main goal of this research is to introduce and develop the necessary features to enhance ADS-IDAC for more practical and realistic applications and to analyze highly dynamic and complex accident scenarios in support of conventional PRAs. This was achieved through the following:

1. Integrating support system FT models into the dynamic simulation runs and tracing their impact on the frontline systems by developing and implementing algorithms for incorporating binary logic of system failures into the dynamic branching rules of the DDETs.
2. Developing and implementing a simplified version of the hybrid response mode to replicate a knowledge-based procedure-following response mode.

3. Introducing a set of comprehensive quantification rules to enable dynamic calculation of branch probabilities and complete risk scenario probabilities. The HFE dependencies were explicitly accounted for through the shared PSFs using a newly developed dynamic Bayesian network (DBN) starting from a BBN model of PSFs developed in the Phoenix method.
4. Selecting appropriate probability distributions for the HFE and creating the capability to select various probability distributions for the hardware failure events, and implementing algorithms to propagate the uncertainties through the DDET.
5. Developing graphical software enhancements to support the previous and current tasks to the ADS-IDAC graphical user interface.
6. Describing and demonstrating why, when, and how ADS-IDAC can be used for dynamic ASP analysis studies.

1.3 Structure of the dissertation

In chapter 2, brief overviews of the ADS-IDAC and HCL methodologies are given to present the state of the art.

In chapter 3, the ADS-IDAC simulation engine and graphical user interface software developmental challenges and solutions necessary to implement and achieve the objective of this research are described.

In chapter 4, the new quantification features implemented into the HCL library are described. These features include multiple types of basic event quantification models, sampling methods, and importance measures.

In chapter 5, the dynamic linking of FTs modeling is introduced and exemplified for capturing the impact of support system failures on the frontline systems.

In chapter 6, a new quantification model for human error is described. It was implemented into ADS-IDAC using a DBN of the crew failure modes and the PSFs. This opened the path for ASD-IDAC to fully quantify the generated DDET with both human and hardware failure events.

In chapter 7, the new branching and quantification of events based on the newly developed dynamically linked FTs for frontline and support systems and DBN for human events are introduced. The techniques used to propagate the aleatory uncertainties through the DDET are described.

Chapter 8 includes a tests case capturing NPP and crew behavior given a concurrent steam generator tube rupture (SGTR) and a main steam line break designed for the international HRA empirical study (Lois, 2009) showcasing the new quantification models implemented into ADS-IDAC.

In chapter 9, a high-level guidance on when and why ADS-IDAC is a good simulation tool to perform an ASP analysis is provided. This guidance is intended to be the starting point in performing an ASP analysis. Appendix 2 contains a full and comprehensive guide to construct an ADS-IDAC simulation model. A test case based on a real accident precursor is given to showcase the new dynamically linked FTs for modeling the impact of support system failures on the frontline systems.

2 Overview of ADS-IDAC and HCL methodologies

This chapter contains brief overviews of the ADS-IDAC and HCL methodologies describe the state of the art.

2.1 Overview of the ADS-IDAC simulation platform

To safely operate NPPs, the crew is required to closely interact with the system – especially under abnormal conditions. Although the NPPs are equipped with automated safety systems, the crew still need to perform complex activities during highly dynamic accident conditions. Their performance could be greatly impaired by the inability to obtain information about the systems or the lack of time available to safely recover the NPP. The safety of NPPs can be improved by predicting and mitigating the conditions that could result in the crew making inappropriate decisions based on conflicting information or commit erroneous actions. Conventionally, the methodologies and techniques used to predict the HFEs are mainly static in time and include limited information about the cognitive context in which the crew perform.

As in the accident at Three Mile Island (TMI) Unit 2 in 1979, the crew can fail to make the correct situational assessment of the NPP by inadequately perceiving the critical system information, and executing the wrong recovery strategy or bypassing important safety systems (Kauffman, 1995). Although no similar accident has occurred since 1979 in the United States, the challenge of correctly predicting and mitigating the human performance remains. The ADS-IDAC model (as discussed) attempts to resolve this challenge. It was developed to tackle the shortcomings of the previous generation of human reliability analysis methods (e.g. THERP), which include the inability to integrate the so-called 'errors of commission'. ADS-IDAC was and remains revolutionary for being the only simulation-based HRA method that not only attempts to quantify the time-dependent crew behavior, but also transparently predict it by simulating both the crew and NPP behavior. It is one of the most mature dynamic platforms with an evolution that spans more than 20 years as illustrated in Figure 2-1. As highlighted in chapter 1, ADS-IDAC is a discrete dynamic event platform.

ADS-IDAC is a simulation engine that includes a scheduler module, a hardware reliability model, an indicator module (the control panel), and the IDAC operator response model coupled with the RELAP5/MOD3.2 thermal hydraulic code (the system model) to generate DDETs containing contextually rich scenarios that could occur given an initiating event. Its modular structure and the flow of information between modules are shown in Figure 2-2. A scheduler module coordinates the interactions between all the other modules and generates the DDETs. The probability of each scenario/sequence is calculated as the product of conditional probabilities of its constituent branches (as is the case for conventional ETs). The indicator module simulates the control panel indicators' states driven by information from the system module. The hardware reliability module simulates the failure probabilities of the system's and control panel's components.

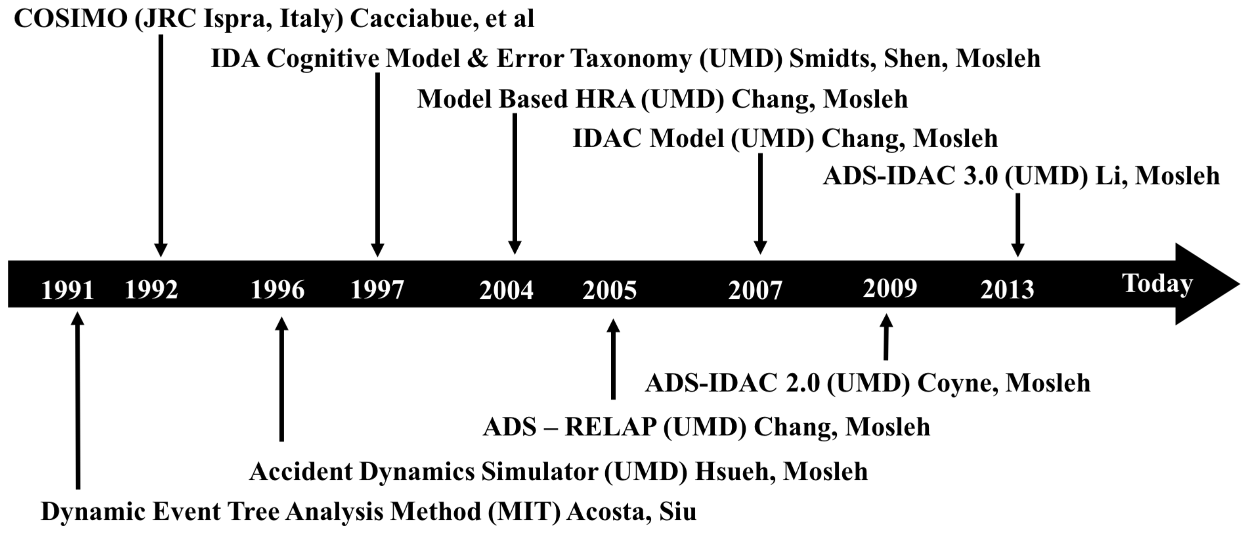

COSIMO (JRC Ispra, Italy) Cacciabue, et al
IDA Cognitive Model & Error Taxonomy (UMD) Smidts, Shen, Mosleh
Model Based HRA (UMD) Chang, Mosleh
IDAC Model (UMD) Chang, Mosleh
ADS-IDAC 3.0 (UMD) Li, Mosleh
1991
1992
1996
1997
2004
2005
2007
2009
2013
Today
ADS-IDAC 2.0 (UMD) Coyne, Mosleh
ADS – RELAP (UMD) Chang, Mosleh
Accident Dynamics Simulator (UMD) Hsueh, Mosleh
Dynamic Event Tree Analysis Method (MIT) Acosta, Siu

Figure 2-1 Development history of ADS-IDAC

The RELAP5 system code is a light water reactor (LWR) best estimate transient simulation tool for modeling the behavior of the reactor coolant systems (RCSs) and the core during postulated events such as loss-of-coolant accidents (LOCAs), loss of offsite power (LOOP), loss of feedwater (LOFW), or loss of flow operational transients (RELAP5, 1995). Its generic modeling approach allows great flexibility in building a various range of thermal hydraulic systems including secondary system components such as turbines, condensers or secondary feedwater systems. Interactive controls and instrumentation allows interaction with the system during the simulation, facilitating an interface for an externally coupled model, such as an operator behavior model.

The mass, momentum, and energy equations are mostly approximated by one-dimensional models limiting the integration of nonlinear or multidimensional effects with a few of exceptions (e.g. cross flow effects in a Pressurized Water Reactor (PWR) core and reflood are modeled using a two-dimensional conduction solution). Great care is advised when judging whether the results offer a reasonable approximation of the physical components and processes being simulated. For ADS-IDAC applications, the validated RELAP5 models reasonably predict the system component interactions and all significant operator interactions with the NPP. The most used plant model in ADS-IDAC is a three-loop PWR with over 250 hydraulic components, 100 heat structures, and 1500 control systems covering all major components and controls referenced in its EOPs.

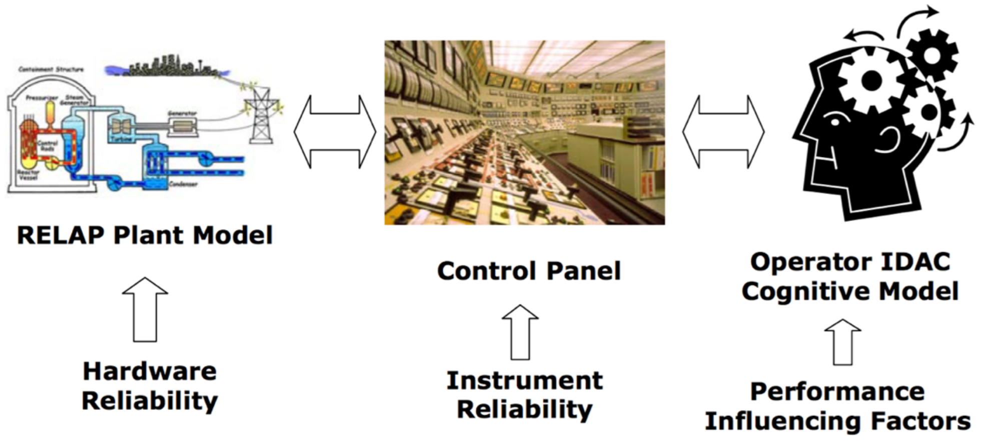


Figure 2-2 ADS-IDAC modules and their information exchange (Coyne, 2009)

Through the hardware reliability model, ADS-IDAC allows the analyst to simulate two types of hardware failure events: (1) time dependent failures, and (2) conditional demand failures. Time dependent failures allow the analyst to initiate hardware state changes (including failures) at a prescribed time during the simulation. Time dependent failures can include hardware failure events, such as failure of pump or a turbine or reactor trip, or may also activate accident initiators included in the RELAP5 thermal hydraulic model such as losses of reactor coolant or steam line breaks. Conditional failures are triggered when a specified component changes its operating state. For example, activation of the reactor trip alarm can be used to generate a conditional failure of an auxiliary feedwater pump. Time dependent failures generate only a single failure event sequence branch while conditional failures generate two event sequence branches – a success path and a failure path. However, both failure types permit the operators to attempt to recover from the failure. If the operator attempts to recover failed equipment, additional sequence branches representing component recovery and permanent failure are generated. Thus, each conditional failure event can result in one of three outcomes: (1) the equipment does not fail, (2) the equipment initially fails but is later recovered, and (3) the equipment fails and is not recovered. Time dependent failures result in two possible outcomes: (1) the equipment initially fails but is later recovered, and (2) the equipment fails and is unrecovered.

The ADS-IDAC simulation model requires either the heuristic cognitive engine or the reasoning machine to guide operator decision-making. The cognitive engine forms a situational assessment from perceived information, to identify and select suitable goals based on the situational assessment, to identify and select suitable strategies for obtaining goals, and to prioritize and resolve conflicts among the selected goal/strategy sets. (Coyne, 2009) The cognitive engine is based largely on a recognition primed naturalistic decision-making model. (Klein, 1997) The recognition primed naturalistic decision-making model, mainly used to simulate the behavior of experienced decision makers under time pressure, was enhanced to capture the variability in human decision-making. As an example, variation in timing of operator response was included through stochastic models. In the reasoning module, the operators' knowledge-based behavior is greatly augmented through an embedded attention mechanism in information perception channels, better capturing cognitive resource limitations and top-down attention control. (Li, 2013) This module also simulates an operator 'making sense' of perceived information, connecting different pieces of information to form a big mental picture of the NPP situation, and making accident diagnoses.

At a high level of abstraction, IDAC is composed of models for information pre-processing, decision-making, and action execution of a crew. Given incoming information, the crew model generates a probabilistic response, linking the context to the action through explicit causal chains.

The following components were developed for the operator heuristic cognitive engine: information pre-processing, decision-making, mental state as the engine of cognition, and action execution module. Together with the reasoning module, they constitute the main parts of the IDAC model shown in Figure 1-4.

The information pre-processing module is one of the key features of the IDAC simulation model. It is based on the premise that all crew behaviors arise from passively perceiving information rather than the direct output from the NPP thermal-hydraulic system model. Moreover, all the information from the outside world (e.g., system, other crew members, etc.) is not registered until it goes through the operator's perception filter. For this reason, the crew's perception filter can act as a screen that either distorts or completely blocks information obtained from the system model. Because of this mechanism, the crew can lead to diagnoses based on distorted or incomplete information and error-forcing contexts. This is one of the goals of ADS-IDAC. Depending on the accident conditions, the resolution of the information pre-processing filter was designed to vary in order to more realistically model the variability in crew performance.

The decision-making module is another important ingredient of the IDAC model. The crew can diagnose and develop strategies for recovery either by using the heuristic cognitive engine, by following the written procedures or using their reasoning capabilities. The procedure-following capability of IDAC accurately represents the written EOPs. Together with the mental state containing the memory management and goal prioritization in the form of mental beliefs, the decision-making module constructs situational assessments through a diagnosis process based on both symptomatic and topographic search processes. The symptomatic search is used to identify memorized events through training that match the observed symptoms, while the topographic search strategy is used to identify the differences between the current system condition and the operator's mental representation of normal operation. A fuzzy logic approach was implemented into the diagnosis module to perform a symptomatic feature matching process. To integrate a topographic search process, a mental representation of the NPP was developed using the mass and energy conservation laws. The diagnosis module also contains heuristic rule sets, goals, strategies, and actions that are used to represent the operator behavior. During the simulation, their activation depends on the operator's situational assessment. An example of a rule is "prevent overfill of the RCS by reducing makeup flow." (Coyne, 2009) Under normal operating conditions, this rule can be routinely applied by the crew to prevent an unnecessary loss of pressure control and a challenge to NPP safety relief valves. Nevertheless, during the Three Mile Island accident, application of this rule led directly to a core melt event because of the erroneous understanding of NPP conditions. It is important to note that using a set of rules to represent human behavior can be unreasonable under novel conditions not covered in the rules. This is a gap that the reasoning module fills although it has the same purpose as the heuristic cognitive engine. By using the reasoning module, the crew are able to create explanations based on their observations collected through a top-down attention control mechanism and using a knowledge web of accident event schemas. Overall, the reasoning module improves the realism of IDAC in modeling operator knowledge, skills, and problem-solving styles.

To include the context and its influence on the operator behavior, static and dynamic PSFs were implemented in IDAC. The static PSFs characterize the fixed crew attributes, while the dynamic PSFs mirror the normal or abnormal conditions of the NPP.

The action execution module is the last module of the IDAC model. This module is used to give the crew the capability to interact with the NPP and change its behavior. The types of actions and their timing depends on their selected recovery strategy. For example, they can choose to recover the NPP using a specific written EOPs. Thus they need to follow the actions specified at each step.

To simulate multiple scenarios under the similar conditions, branching rules are included to reflect the variability in crew and system behavior. They represent slow or fast procedure execution speed, skipping of procedure steps, reliance on memorized information, activation of mental beliefs, variations in control inputs, and equipment failures. The implementation of a NPP functional decomposition and diagnostic engine strengthened the ability to model knowledge-based actions and other cognitive feature such as procedure step-skipping.

Overall, ADS-IDAC is a very powerful simulation platform that is capable of modelling NPP and crew behavior, including their interaction given abnormal conditions. From previous studies, the following conclusions can be made:

- ADS-IDAC can simulate the direct effect of crew behavior on the NPP recovery given abnormal conditions.
- ADS-IDAC can model the decision-maker's underlying cognitive processes.
- ADS-IDAC can model the variability in crew decisions and behaviors by using a relatively small number of branching rules.

2.2 Overview of the HCL methodology

The HCL methodology (Wang, 2007) was developed for risk scenario analysis in PRAs of technological systems that considers not only the risks associated with hardware components (also called 'hard' causes), but also the risks generated by human activities, physical environment or socio-economic environment (also called 'soft' causes). This methodology offers a multi-layered modeling approach so that each individual domain of the system is modeled with the most appropriate technique. The three layers modeled in HCL are:

1. The ESD layer - it is used to model the risk context

2. The FT layer – it is used to model the physical systems' behavior and quantify their impact on their corresponding linked events in the ESD.
3. The BBN layer – it is used to model the causal relations between events that have 'soft' root causes.

The Hybrid Causal Logic (HCL) methodology is a static multi-layered modeling approach that allows the most appropriate modelling techniques to be applied to different domains of the system. It is an integrated model with the corresponding sets of taxonomies, analytical and computational procedures. (Wang, 2007).

The combinations of ESDs, FTs, and BBNs are defined as HCL diagrams. In an HCL diagram, BBNs can be used to model the basic events in the FTs or the top events in the ESDs. The BBNs can also have common nodes. The combination can be made at any level to build an HCL diagram.

One of the most effective ways of modeling risks associated with complex systems is by using ESDs as the first layer of describing system behavior in case of abnormal conditions, and then FTs as a second layer of detailing the contributing causes of the events in the ESDs. HCL adds a third layer of modeling through BBNs to better represent the common cause failures and soft causal dependencies stemming from the socio-economical, regulatory or physical environment.

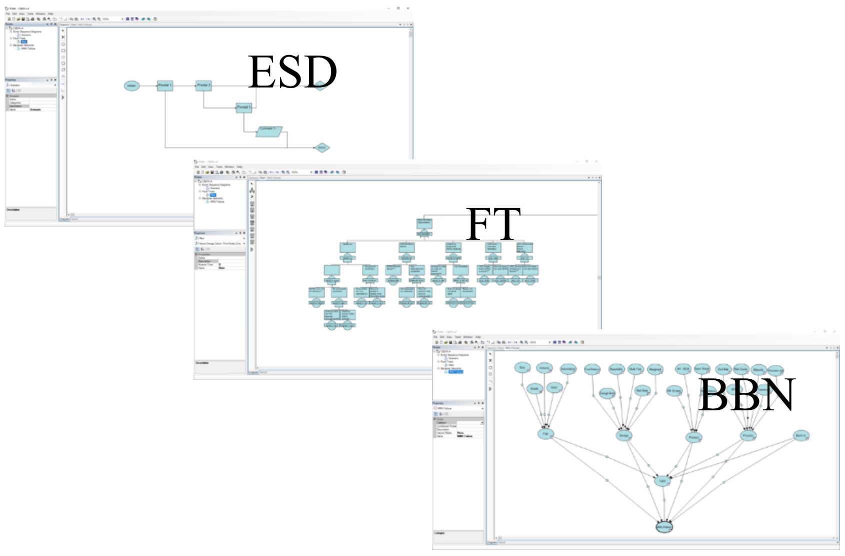


Figure 2-3 IRIS graphical user interface screenshots of the ESD, FT, and BBN layers

The FT analysis layer is used to model a system's failure mechanisms by decomposing them into its subsystems based on the functional interaction between them using simple logic gates (e.g. OR, AND, K-out-of-N). Given that the basic events in FTs are often binary and that the FT structure is a Boolean expression, the HCL methodology contains algorithms to convert the FTs to an equivalent binary decision diagram.

A binary decision diagram is a binary decision tree over the Boolean variables where identical Boolean expressions are unified. In graph theory, a binary decision diagram is a rooted, directed, acyclic graph with an unconstrained number of in-edges and two out-edges, one for each two states (i.e. true or false) of any given variable. Thus, the binary decision diagram has two terminal nodes labeled 0 (false) and 1 (true) representing the final value of the logical expression. Various FT logical gates are shown with their equivalent binary decision diagram representations in Figure 2-4.

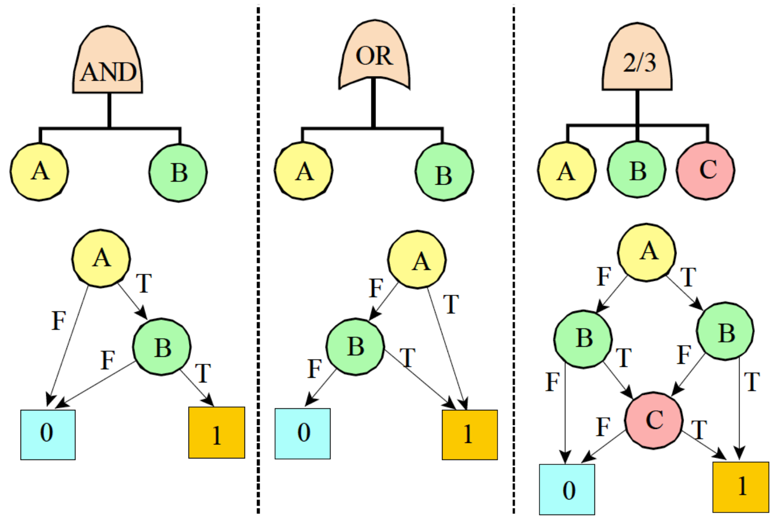


Figure 2-4 Binary decision diagram representations of FTs

The ESD analysis layer can also be converted to a binary decision diagram due to its binary logic structure. The conversion of ESDs and FTs to a single combined binary decision diagram and subsequence solution has proven to be an efficient way to evaluate system state probabilities (Groen, 2006).

The BBN analysis layer provides an efficient and realistic graphical representation of causal relations between elements of a domain under investigation. BBNs are as user-friendly as commonly used FTs. The simple graphical structure shows properties of the problem domain in an intuitive way, which also makes it easy for BBNs 'non-experts' to understand and build them based on either expert or empirical knowledge. Finally, inference algorithms can be applied to BBNs for accurately propagating new evidence when it becomes available, and updating the probabilities of conditional events.

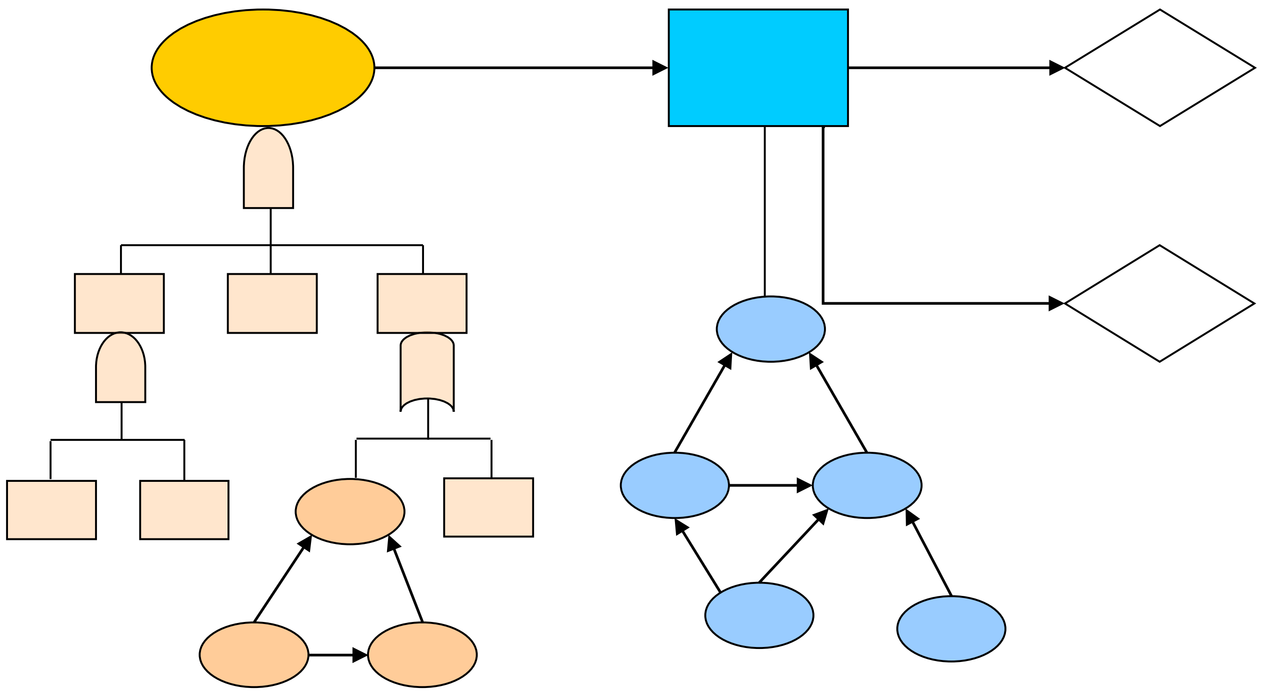


Figure 2-5 Event sequence diagram with linked FT and BBN (Mosleh, 2015)

The overall HCL quantification algorithm is built on the binary decision diagram methodology. All three layers — ESDs, FTs, and BBNs — are included (Figure 2-5). The HCL's binary decision diagram solver is based on the IF-THEN-ELSE (Rauzy, 1997) algorithm that includes the negated part. This means the common (intersectional) nodes, which are both in the BBN and FTs or ESDs, can be well quantified.

A binary decision diagram is created for each scenario from the corresponding ESD during quantification. Each scenario has the typical structure starting with an initiating event, followed by a sequence of occurrences and negations of branching events, and an end state. The conditions and probabilities under which the initiating event and branching events occur are specified with the aid of FTs or BBNs that are also converted to binary decision diagrams. Eventually, the binary decision diagram representing the full scenario is found by merging the binary decision diagrams obtained from the FTs and BBNs describing the initiating event and branching events according to the logic of the ESD.

Most of the computation time is spent on converting the FTs and BBNs to binary decision diagrams and on the subsequent ordering and reduction of the binary decision diagram to obtain a reduced ordered binary decision diagram. This computational cost is offset by the relatively costlier and less accurate conventional solution algorithms that find the minimal cut sets as an intermediate stage. The final step in the quantification solution necessitates calculating each sequence probability that is a function of the probabilities of its constituent events. The Big-O time complexity of this step is O(n), as it requires only a binary decision diagram traversal where the Shannon decomposition is applied at each node.

3 Software developmental work on the ADS-IDAC simulation platform

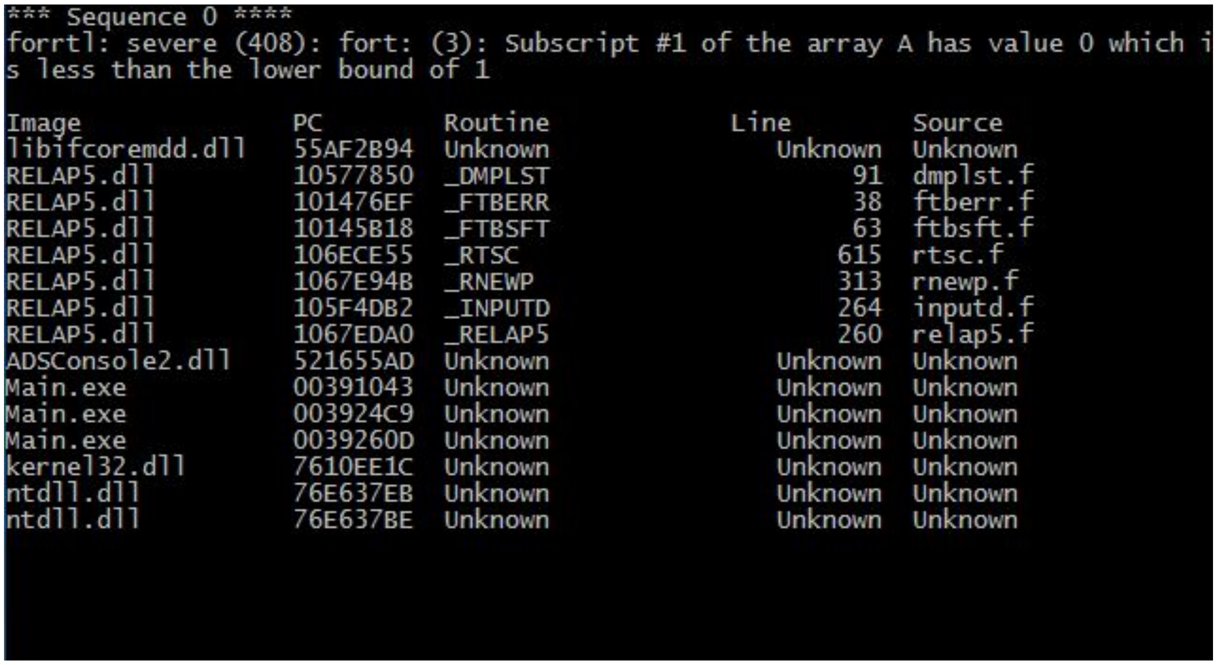
In this chapter, the ADS-IDAC simulation engine and graphical user interface software developmental work necessary to implement and achieve the objective of this research is described. Using the C++ engine and Java graphical user interface with the latest compilers presented many challenges that are briefly described together with their solution in the following subsections.

3.1 ADS-IDAC engine developmental enhancements

The current ADS-IDAC engine is based on C++ ADS-IDAC 3.0, which was internally coupled with the Fortran RELAP5/MOD3.2.2. They were compiled on Windows in the Visual Studio Environment with the VS C++ and VS Fortran compiler 6.0 using a Fortran/C++ interface with shared variables for data exchange and a few interface functions for transition between RELAP and ADS-IDAC at each time step.

Initially, much effort was spent on trying to compile and build the ADS-IDAC 3.0 in Visual Studio 2014 with the Intel C++ and Fortran compilers. Many errors and issues related to the new compilers and the new building process for mixed languages of the Visual Studio Environment were fixed. The source code was separated into one Fortran and one C++ library which were dynamically linked because of the cyclic dependencies.

The simulation engine executable was successfully built. However, while running the simulation engine, an error showed up in RELAP5 that even with the efforts from the RELAP5 support team could not be resolved (Figure 3-1). To overcome the RELAP5 error, the latest stable version (RELAP5/MOD3.3) was tested and validated with the latest compilers. This update required substantial effort to recouple, test, and validate RELAP5 with ADS-IDAC. Eventually, this effort led to the first successful simulation run of ADS-IDAC 3.0 with the latest Intel C++ and Fortran compilers.



```
*** Sequence 0 ****
forrtl: severe (408): fort: (3): Subscript #1 of the array A has value 0 which i
s less than the lower bound of 1

Image              PC        Routine            Line        Source
libifcoremdd.dll   55AF2B94  Unknown               Unknown  Unknown
RELAP5.dll         10577850  _DMPLST                    91  dmplst.f
RELAP5.dll         101476EF  _FTBERR                    38  ftberr.f
RELAP5.dll         10145B18  _FTBSFT                    63  ftbsft.f
RELAP5.dll         106ECE55  _RTSC                     615  rtsc.f
RELAP5.dll         1067E94B  _RNEWP                    313  rnewp.f
RELAP5.dll         105F4DB2  _INPUTD                   264  inputd.f
RELAP5.dll         1067EDA0  _RELAP5                   260  relap5.f
ADSConsole2.dll    521655AD  Unknown               Unknown  Unknown
Main.exe           00391043  Unknown               Unknown  Unknown
Main.exe           003924C9  Unknown               Unknown  Unknown
Main.exe           0039260D  Unknown               Unknown  Unknown
kernel32.dll       7610EE1C  Unknown               Unknown  Unknown
ntdll.dll          76E637EB  Unknown               Unknown  Unknown
ntdll.dll          76E637BE  Unknown               Unknown  Unknown
```

Figure 3-1 ADS-IDAC 3.0 error in the RELAP5 library

Given the size of the source code, this setup led to build times exceeding 30 minutes. This would have been unacceptable during the development process, as any change would have required a 30-minute wait time. No new features could have reasonably been implemented given the long building time. The solution turned out to be two-fold:

- The C++ dynamic library was split further into smaller static libraries based on their function to reduce the dependencies between them: “branch”, “calculationaid”, “controlpanel, “crew”, “global”, “hwreliability”, “procedure”, “reasonmachine”, “scheduler”, and “system”. In this way, any change inside one of the classes requires the recompilation and rebuilding of only the library that contains it and the recompilation of its dependent classes.
- The Visual Studio Environment was replaced with CMake (https://cmake.org) for the building process. This introduced the potential to make ADS-IDAC cross-platform and

relieved the developers from updating and configuring the compilation process inside Visual Studio manually that can potentially lead to linking issues.

Currently, any change in one of the source code files requires less than a minute for recompiling and rebuilding the executable, thereby reducing the development time considerably. ADS-IDAC was successfully compiled and built for the first time on Ubuntu and macOS. Other UNIX platforms are expected to be compatible, however they have not been tested. CLion IDE (https://www.jetbrains.com/clion/), which supports C++11 natively and uses CMake as a project model, was adopted for the developing environment and recommended for future development. Given the flexibility of CMake, future developers can use any default development environment available on their system. Appendix 1 includes the platform dependent requirements and a guide for compiling and building the ADS-IDAC executable from the source code.

One of the new features that was implemented into the simulation engine was support systems integration by dynamically linking FTs. This was achieved by linking the "controlpanel" library to the "HCL" library developed based on the HCL methodology (Figure 3-2). In turn, this allowed the frontline systems to be modeled with dynamic FTs. Therefore, through internal coupling the HCL library became integrated into the ADS-IDAC. The linking required updates to the HCL library compilation process like those changes done to the ADS-IDAC source code. Thus, the initial development effort focused on making the HCL library cross-platform and updating the compilation process with CMake. The upgraded HCL library has been successfully tested on macOS 10.12, Windows 10, and Ubuntu 16.

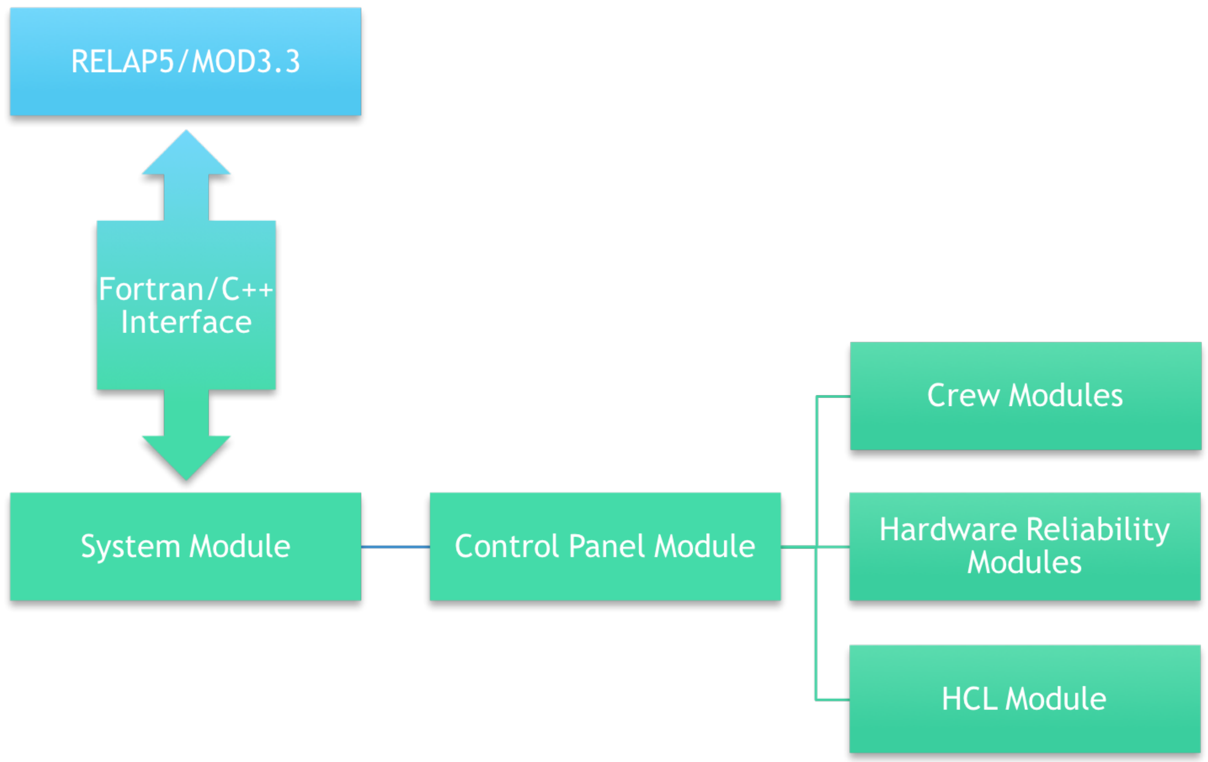


Figure 3-2 ADS-IDAC Modular Interaction

This coupling gave access to HCL's capability for creating, updating and quantifying of FTs. Also, HCL supports creating, updating and evaluating of BBNs. A BBNs does not require complete knowledge of the cause-and-effect relations between the random variables. HCL offers a natural platform for creating a BBNs for all the PSFs when more empirical data becomes available. The modeling of the impact of the situational and cognitive factors on the operator's behavior is greatly improved.

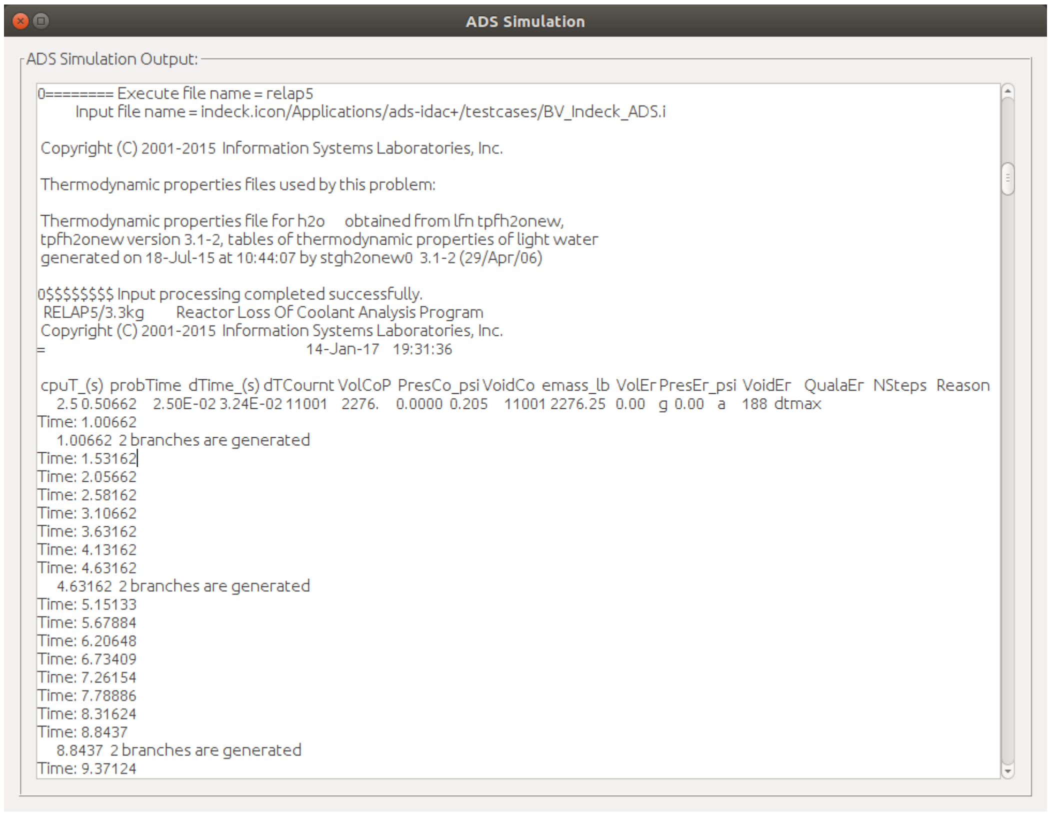
However, both ADS-IDAC and the HCL library had to be expanded to fully implement the support systems integration with new hardware reliability features, a BBN for HFEs, and uncertainty propagation, even if the coupled HCL library was used as the basis for this work. This work is further detailed in the following chapters.

3.2 ADS-IDAC graphical user interface development

Some of the new features associated with Java graphical user interface developmental updates are listed below:

- Cross-compatibility: The graphical user interface has been updated and tested to run on Windows, mac OS X, and Ubuntu. The updates accommodated the graphical user interface for the different file systems on each environment and included a simulation engine built for each platform.
- Simulation engine updates: All the updates on the simulation engine performed during this research required updates in the graphical user interface. This included the rebuilding of the simulation engine executables. Also, updates were implemented for the creating of new data, graphical objects, and their controllers for the user-friendly input and output of information compatible with the simulation engine requirements.
- Multithreaded running of the simulation engine: To prevent the graphical user interface from freezing while the simulation engine is running, the execution of the simulation engine had to be moved on a separate thread.
- Capability to run the engine from inside the graphical user interface: Java uses JAR package files formats that contain all the classes, metadata, and resources into one file necessary for distribution. Unfortunately, due to Java restrictions an executable placed inside the JAR package cannot be run. One way to overcome this restriction is to distribute the external executable (which in this case was the simulation engine) outside the JAR package and to ask the user to set the path to the executable for each installation. However, this solution expects the user to perform an extra step that can lead to other issues. Therefore, the simulation engine executable was included in the JAR package together with a script that contains instructions to copy the executable to the default temporary folder location, to make that path known to the Java graphical user interface, and after running the external executable to perform the simulation, to remove the external executable from the temporary folder. Each time a simulation is performed, the script is running automatically in the background without requiring any interaction from the user.

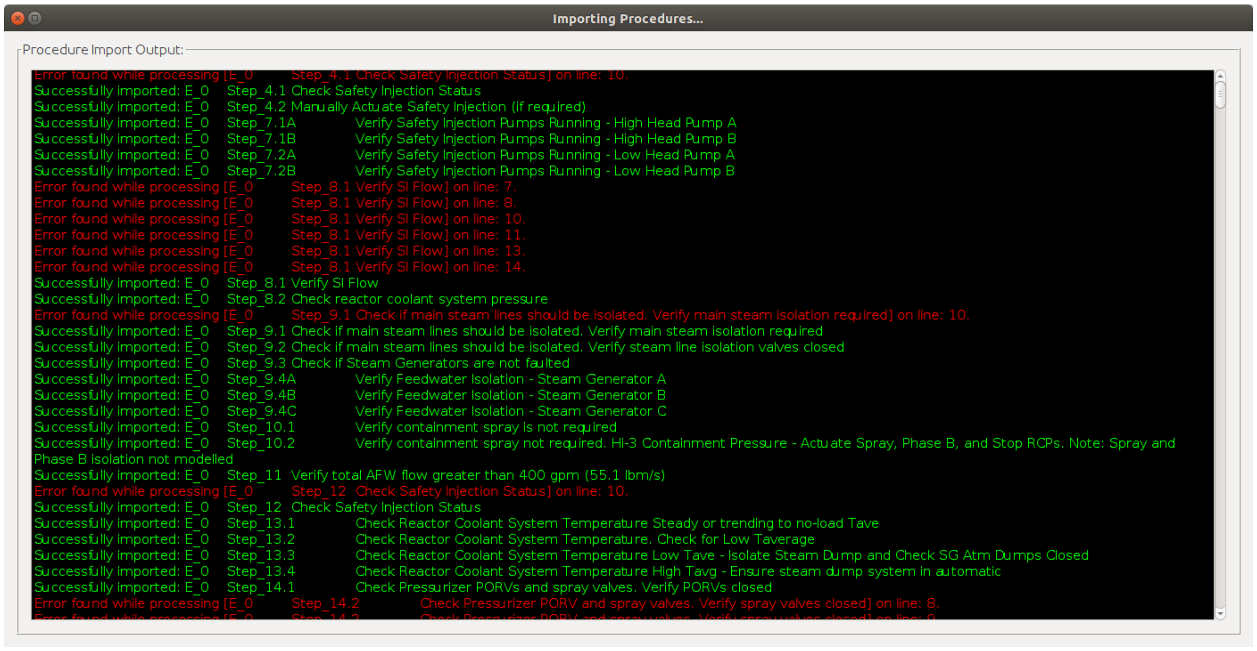
- Buffering output stream and safely stopping the simulation: While the simulation is running, a continuous stream of information is outputted both to files and to an output window making the progress of the simulation visible to the user (Figure 3-3). In the background, the simulation engine is generating a high flux of information into an output stream that has a limited capacity. This stream of information must be read by the graphical user interface continuously to prevent it from overfilling. It is possible for the graphical user interface to not be able to process and output the information into the window fast enough given certain machine configurations. To avoid this issue, an algorithm was implemented that essentially buffers the output stream, thus allowing the graphical user interface to process the information at its own speed. An added benefit of this update is the ability to implement a feature to safely stop the simulation at any time during its execution.



```
0======== Execute file name = relap5
      Input file name = indeck.icon/Applications/ads-idac+/testcases/BV_Indeck_ADS.i

 Copyright (C) 2001-2015  Information Systems Laboratories, Inc.

 Thermodynamic properties files used by this problem:

 Thermodynamic properties file for h2o     obtained from lfn tpfh2onew,
 tpfh2onew version 3.1-2, tables of thermodynamic properties of light water
 generated on 18-Jul-15 at 10:44:07 by stgh2onew0  3.1-2 (29/Apr/06)

0$$$$$$$$ Input processing completed successfully.
 RELAP5/3.3kg       Reactor Loss Of Coolant Analysis Program
 Copyright (C) 2001-2015  Information Systems Laboratories, Inc.
=                                        14-Jan-17   19:31:36

 cpuT_(s) probTime  dTime_(s) dTCournt VolCoP  PresCo_psi VoidCo  emass_lb  VolEr PresEr_psi  VoidEr   QualaEr  NSteps  Reason
   2.5 0.50662    2.50E-02 3.24E-02 11001    2276.    0.0000  0.205    11001 2276.25  0.00    g  0.00    a    188  dtmax
Time: 1.00662
   1.00662  2 branches are generated
Time: 1.53162
Time: 2.05662
Time: 2.58162
Time: 3.10662
Time: 3.63162
Time: 4.13162
Time: 4.63162
   4.63162  2 branches are generated
Time: 5.15133
Time: 5.67884
Time: 6.20648
Time: 6.73409
Time: 7.26154
Time: 7.78886
Time: 8.31624
Time: 8.8437
   8.8437  2 branches are generated
Time: 9.37124
```

Figure 3-3 Streaming simulation output in the graphical user interface from the simulation engine.

- Algorithm to batch import and validate procedures: The task of creating the procedures is very time consuming and error prone. An importing and validating algorithm was implemented that greatly speeds up the process when previously written ADS-IDAC procedures are available. Errors during processing are highlighted in red, and the user manual should be used for correcting them (Figure 3-4).



Importing Procedures...

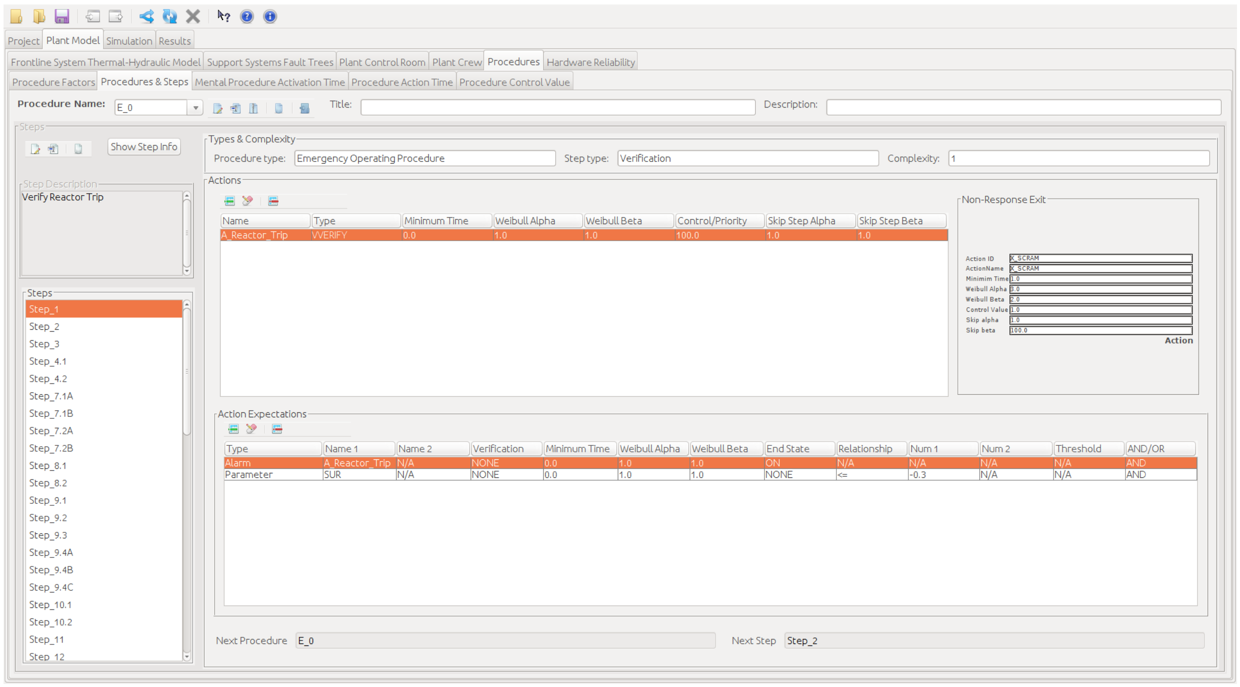
Procedure Import Output:

```
Error found while processing [E_0       Step_4.1 Check Safety Injection Status] on line: 10.
Successfully imported: E_0    Step_4.1 Check Safety Injection Status
Successfully imported: E_0    Step_4.2 Manually Actuate Safety Injection (if required)
Successfully imported: E_0    Step_7.1A           Verify Safety Injection Pumps Running - High Head Pump A
Successfully imported: E_0    Step_7.1B           Verify Safety Injection Pumps Running - High Head Pump B
Successfully imported: E_0    Step_7.2A           Verify Safety Injection Pumps Running - Low Head Pump A
Successfully imported: E_0    Step_7.2B           Verify Safety Injection Pumps Running - Low Head Pump B
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 7.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 8.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 10.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 11.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 13.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 14.
Successfully imported: E_0    Step_8.1 Verify SI Flow
Successfully imported: E_0    Step_8.2 Check reactor coolant system pressure
Error found while processing [E_0       Step_9.1 Check if main steam lines should be isolated. Verify main steam isolation required] on line: 10.
Successfully imported: E_0    Step_9.1 Check if main steam lines should be isolated. Verify main steam isolation required
Successfully imported: E_0    Step_9.2 Check if main steam lines should be isolated. Verify steam line isolation valves closed
Successfully imported: E_0    Step_9.3 Check if Steam Generators are not faulted
Successfully imported: E_0    Step_9.4A           Verify Feedwater Isolation - Steam Generator A
Successfully imported: E_0    Step_9.4B           Verify Feedwater Isolation - Steam Generator B
Successfully imported: E_0    Step_9.4C           Verify Feedwater Isolation - Steam Generator C
Successfully imported: E_0    Step_10.1           Verify containment spray is not required
Successfully imported: E_0    Step_10.2           Verify containment spray not required. Hi-3 Containment Pressure - Actuate Spray, Phase B, and Stop RCPs. Note: Spray and
Phase B isolation not modelled
Successfully imported: E_0    Step_11  Verify total AFW flow greater than 400 gpm (55.1 lbm/s)
Error found while processing [E_0       Step_12  Check Safety Injection Status] on line: 10.
Successfully imported: E_0    Step_12  Check Safety Injection Status
Successfully imported: E_0    Step_13.1           Check Reactor Coolant System Temperature Steady or trending to no-load Tave
Successfully imported: E_0    Step_13.2           Check Reactor Coolant System Temperature. Check for Low Taverage
Successfully imported: E_0    Step_13.3           Check Reactor Coolant System Temperature Low Tave - Isolate Steam Dump and Check SG Atm Dumps Closed
Successfully imported: E_0    Step_13.4           Check Reactor Coolant System Temperature High Tavg - Ensure steam dump system in automatic
Successfully imported: E_0    Step_14.1           Check Pressurizer PORVs and spray valves. Verify PORVs closed
Error found while processing [E_0       Step_14.2           Check Pressurizer PORV and spray valves. Verify spray valves closed] on line: 8.
Error found while processing [E_0       Step_14.2           Check Pressurizer PORV and spray valves. Verify spray valves closed] on line: 9.
```

Figure 3-4 Procedures validation output

- Algorithms to import individual '.txt' input files: The ADS-IDAC graphical user interface uses its own '.ads' file format to store the input and output simulation data. Unfortunately, any graphical user interface update that modifies the structure of the data contained in the '.ads' file, implicitly makes the old .ads simulation files incompatible with the new '.ads' file format expected by the graphical user interface. This issue essentially renders the previous '.ads' simulation files unusable and a script that understands every previous version of the graphical user interface would be necessary to recover the data. Fortunately, this is not necessary because the simulation engine stores the information in '.txt' files that do not have such problems. An algorithm was implemented for each '.txt' input file used by ADS-IDAC. This solution still required a substantial amount of developmental effort given the large number of input files. However, it was a necessary task to not only enhance the user-friendliness of the graphical user interface, but also speed up future developmental updates.

- New icon pack and refreshed look and feel: The design of the graphical user interface has been refreshed by improving the layout and replacing the default icon pack with a more expressive icon pack for a simulation environment (Figure 3-5).



Procedure Name: E_0 Title: Description:

Procedure type: Emergency Operating Procedure Step type: Verification Complexity: 1

Step Description: Verify Reactor Trip

Steps: Step_1, Step_2, Step_3, Step_4.1, Step_4.2, Step_7.1A, Step_7.1B, Step_7.2A, Step_7.2B, Step_8.1, Step_8.2, Step_9.1, Step_9.2, Step_9.3, Step_9.4A, Step_9.4B, Step_9.4C, Step_10.1, Step_10.2, Step_11, Step_12

Name	Type	Minimum Time	Weibull Alpha	Weibull Beta	Control/Priority	Skip Step Alpha	Skip Step Beta
A_Reactor_Trip	VVERIFY	0.0	1.0	1.0	100.0	1.0	1.0

Action ID	X_SCRAM
ActionName	X_SCRAM
Minimim Time	1.0
Weibull Alpha	3.0
Weibull Beta	2.0
Control Value	1.0
Skip alpha	1.0
Skip beta	100.0

Type	Name 1	Name 2	Verification	Minimum Time	Weibull Alpha	Weibull Beta	End State	Relationship	Num 1	Num 2	Threshold	AND/OR
Alarm	A_Reactor_Trip	N/A	NONE	0.0	1.0	1.0	ON	N/A	N/A	N/A	N/A	AND
Parameter	SUR	N/A	NONE	0.0	1.0	1.0	NONE	<=	-0.3	N/A	N/A	AND

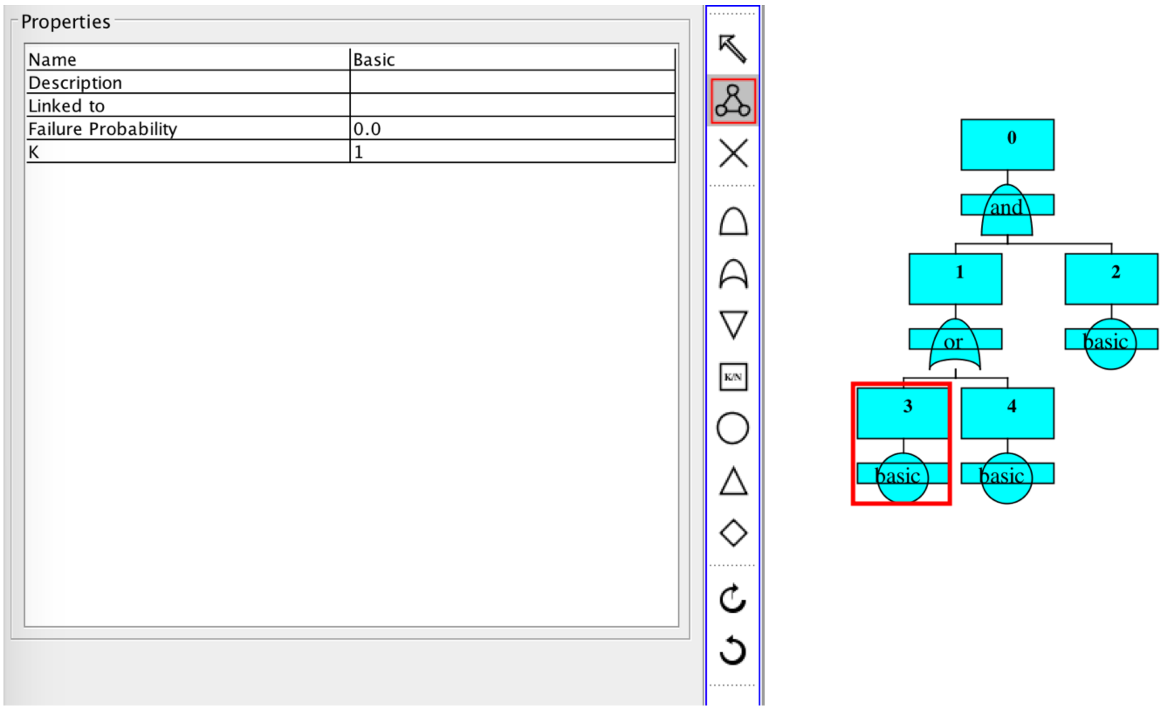
Next Procedure E_0 Next Step Step_2

Figure 3-5 ADS-IDAC graphical user interface look and feel

- Resolved '.txt' input files compatibility issue between the graphical user interface and simulation engine: Some of the '.txt' input files created by the graphical user interface with information from the user were not compatible with the simulation engine. Considerable effort was applied debugging the algorithms that are used to create these files.

- Accommodating input of the reasoning module information: In the ADS-IDAC 3.0 simulation engine Yuandan Li implemented a new operator reasoning capability that requires its own input information in the form of '.txt' files. Thus, all the necessary updates were implemented into the graphical user interface to allow the user to create or import the '.txt' files needed in the reasoning module.

- Create or edit FTs: Yucong Li developed a Java graphical user interface with which FTs can be created and saved as '.xml' format compatible with the HCL library. (Figure 3-6) This feature has been included in the ADS-IDAC graphical user interface to give the users the flexibility to choose between ADS-IDAC or IRIS graphical user interface for creating FTs.


Properties

Name	Basic
Description	
Linked to	
Failure Probability	0.0
K	1

Figure 3-6 Screenshot of graphical user interface for FTs

The graphical user interface has been rigorously tested and debugged, which enabled for the first time the building of an ADS-IDAC simulation model and fully run it inside the graphical user interface without the need to create or edit any '.txt' files. Appendix 2 contains a guide for creating and running a simulation case with the ADS-IDAC graphical user interface.

4 New HCL quantification features

In this chapter, the new features implemented into the HCL library are described. New quantification models were included to allow the modelling of more advanced hardware failures. For uncertainty propagation, a number of sampling methods have been included into the HCL library. A range of importance measures has been implemented that can be used to assess the relative importance of different components in a system modeled with a FT. Finally, the leaky noisy OR gates have been implemented for quantifying BBNs that contain a large number of nodes.

4.1 Basic event quantification models

The following quantification models were implemented into the HCL library to expand the types of system failures that ADS-IDAC can simulate and provide the necessary probability distributions for uncertainty quantification.

4.1.1 Time-dependent models

The open source Boost libraries distributed under the Boost Software License, Version 1.0 [http://www.boost.org/users/license.html] have been linked to the HCL library to leverage their mature statistical and file management capabilities. The distributions for uncertain parameters added to HCL using the Boost libraries and their associated input XML snippets are given in Table 4-1.

Table 4-1 Probability distributions with XML input snippets

Probability Distribution Type	XML input snippet
Uniform distribution	<uniform min="1000.0" max="2000.0" />
Normal distribution	<normal mean="5.0" std="0.1" />
Lognormal distribution	<lognormal mean="0.2" std="0.1" />

Gamma distribution	<gamma shape="1.0" scale="2.0" />
Noncentral chi-squared distribution	<chi k="2.0" lambda="1.0" />
Cauchy distribution	<cauchy mean="1.2" std="0.1" />
Student's t distribution	<student degreesOfFreedom="2.0" />
Weibull distribution	<weibull shape="5.0" scale="1000.0" />

The time-dependent models implemented with their associated XML snippets are given in Table 4-2.

Table 4-2 Time-dependent probability distributions with XML input snippets

Probability Distribution Type	**XML input snippet**
Exponential distribution	<exponential failureRate="0.001" testInterval="100.0" />
Exponential distribution with uncertain failure rate	<uncertainExponential failureRate="normal" testInterval="5000.0"> <normal mean="3.0e-4" std="1.0e-7" /> </uncertainExponential>
Weibull distribution	<weibull shape="0.1" scale="200.0" testInterval="100.0" />
Weibull distribution with	<uncertainWeibull shape="normal" scale="uniform" testInterval="9.0"> <normal mean="1.2" std="0.1" />

uncertain shape and scale	<uniform min="900.0" max="1100.0" /> </uncertainWeibull>

In practice, these are the most commonly used time-dependent distributions and should cover a wide range of applications. If other time-dependent distributions are needed, the HCL library could be easily extended to accommodate them given the modular object-oriented design pattern used in implementing the initial ones.

4.1.2 Nonparametric models

Time-dependent reliability results from third-party advanced models of component failure mechanisms (e.g. MATLAB simulation) may be used in the system analysis. If the results cannot be fitted to the common statistical distributions, they can be included in the analysis in the form of a nonparametric model given by the time-dependent reliability cumulative distribution function for each component. The data should be included in a '.csv' file containing two rows for time and reliability data pairs. The input XML snippet to include the data for a component defined using a nonparametric model is:

```
<nonparametric dataFile="path/to/file/name.csv" testInterval="5000.0" />
```

4.1.3 Expression user-defined models

The first developmental task pursued to create a generic expression parsing module with added distribution recognition and global variable functionalities was to link the C++ Mathematical Expression Parsing And Evaluation Library (called 'exprtk' and available for free use under the MIT license [http://partow.net/programming/exprtk/]).

The exprtk library was extended to include the dictionary of the following commonly used distribution functions: “uniform”, “normal”, “lognormal”, “exponential”, “weibull”. These can be individually used to model uncertain failures on demand or be part of expressions defining complex failure mechanisms including time-to-failure or other user-defined global variables.

For example, temperature could be a global variable for an Arrhenius degradation model. A user could define the temperature to be sampled from a distribution or fix it to a certain value through the assignment operator. In this example, it is assumed the component time to failure is exponentially distributed with the failure rate given by an Arrhenius degradation model. The input XML snippet for this case is:

```
<expression expressionLiteral="T:=uniform(223.2,353.2);
1-exp(-(1/(5.57e-6*exp(8566/T)))*t)" testInterval="500.0"/>
```

In other component definitions that require the use of the same temperature variables, the user would not need to redefine it. In other words, each global variable need be defined only once, otherwise it will be overwritten.

4.2 Uncertainty analysis

The sampling techniques presented here were ultimately used in ADS-IDAC to perform uncertainty analysis on the discrete DDETs.

4.2.1 Forward sampling

Uncertainty quantification requires a sampling strategy. All strategies implemented in the HCL library are what is called ‘forward samplers.’ as they do not learn from information gathered during the sampling of the system (as for example ‘adaptive samplers’ do.)

The HCL library includes Monte Carlo sampling (MCS) and three variants of Latin Hypercube sampling (LHS). The HCL library offers more advanced forward sampling strategies that use low-discrepancy sequences called quasi-Monte Carlo sampling (QMCS). The difference between these sampling methods can be seen in Figure 4-1.

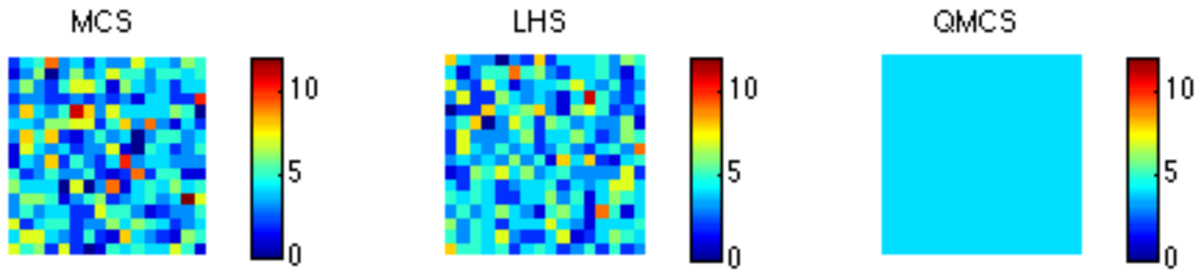


Figure 4-1: Density plot of Monte Carlo sampling (MCS), Latin hypercube sampling (LHS) and quasi-Monte Carlo sampling (QMCS) on a 16x16 grid with 1024 samples (Groen, 2017)

4.2.2 Monte Carlo Simulation

Monte Carlo simulation leverages the law of large numbers (amongs other things) to estimate the expectation using the sample mean of a function of a set of sampled random variables. To initialize the sampler, a priori knowledge of the needed number of samples or the number of dimensions is not required. Forward sampling generates pseudorandom numbers using the Mersenne Twister algorithm without considering the previously generated sample points. Monte Carlo is arguably the most used sampling strategy across multiple fields. (Matsumoto, 1998)

4.2.3 Latin Hypercube Sampling

LHS requires the number of samples at initialization. The number of samples is then used to stratify the domain space into Latin squares such that each sample's location must be remembered to not be explored at future iterations. In two dimensions, a Latin square is a reduced to a square matrix as can be seen in Figure 4-2.

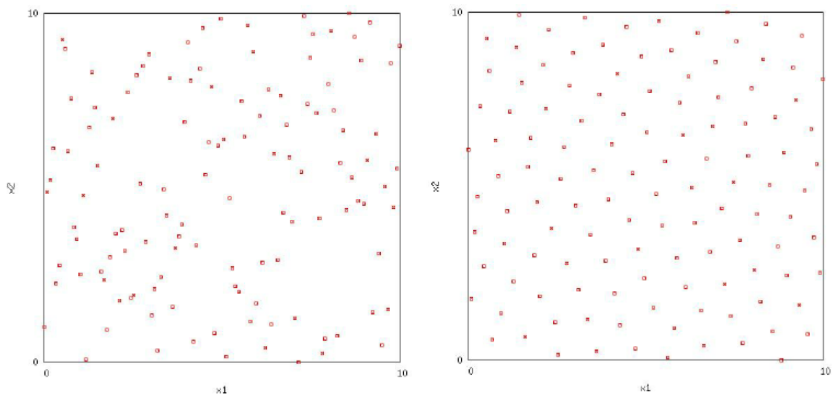


Figure 4-2: Left: Monte Carlo sampling, Right: Latin Hypercube Sampling

LHS Center: After a Latin square is generated, LHS Center determines each subsquare's center that is equidistant from the squares corners or apexes.

LHS Random: After a Latin square is generated, LHS Random determines each subsquare's center that is randomly located within the squares corners or apexes.

Improved Distributed LHS: Based on techniques outlined in Beachkofski and Grandhi (2002), this approach finds a set of samples that are optimally spread out in the domain space as illustrated in Figure 4-3.

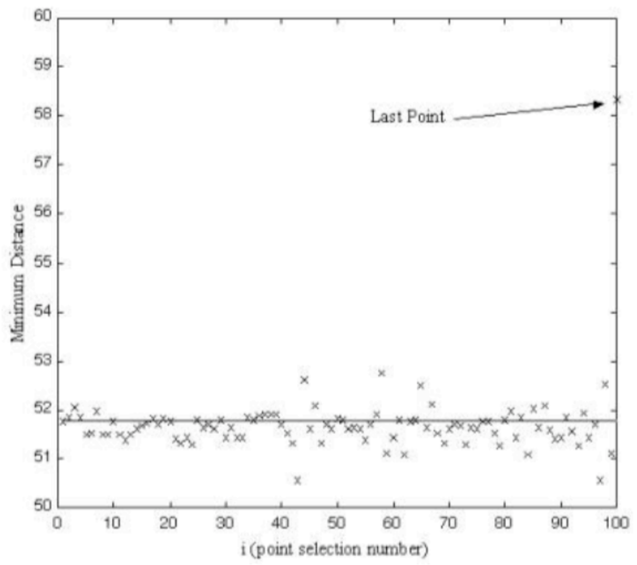


Figure 4-3: Minimum distance between sample points (Beachkofski and Grandhi, 2002)

For problems with many dimensions, Improved Distributed LHS is considerably slower than the other options available in the HCL library as its Big-O time complexity is $O(n^2)$.

4.2.4 Quasi-Monte Carlo Sampling

QMCS employs a quasi-random number generator. A quasi-random or low-discrepancy sequence (such as the Faure, Halton, Hammersley, Niederreiter or Sobol sequences) is less 'random' than a pseudorandom number sequence, but more useful for tasks such as uncertainty quantification in higher dimensions. This is because low discrepancy sequences tend to explore the space more evenly than random numbers as successive samples are generated in a position as far as possible from the previous samples. Low-discrepancy sequences avoid clustering of samples as can be seen in Figure 4-4 for the Sobol sequence.

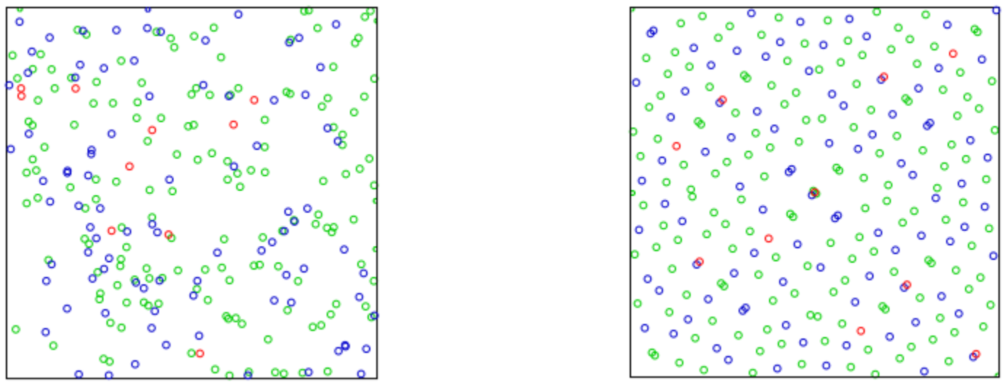


Figure 4-4: Left: A 2-dimensional sequence of pseudorandom numbers, showing the first 10 (red), 100 (red + blue) and 256 (red + blue + green). Right: A Sobol sequence of low-discrepancy quasi-random numbers, showing the first 10 (red), 100 (red + blue) and 256 (red + blue + green) points from the sequence. (By Jheald [CC BY-SA 3.0] via Wikimedia Commons)

All the QMCS are implemented in the HCL library through an interface to the C++ libraries provided by John Burkardt under the GNU LGPL license[1] and use the low-discrepancy sequences described below.

[1] http://people.sc.fsu.edu/~jburkardt/

Halton sequence: The Halton sequence (Halton, 1960) is a generalization of the 1-dimensional van der Corput sequence. (van der Corput, 1935) An n-dimensional Halton sequence contains n 1-dimensional van der Corput sequences that use as bases the first n primes. It is valid up to 1229 dimensions.

Faure sequence: Unlike the Halton sequence, the Faure sequence sets only one base for all dimensions and it reorders the sequence of elements within each dimension. The base of a Faure sequence is the first prime number after or equal to the number of dimensions (Faure, 1982). By permuting the sequence within each dimension, the Faure sequence avoids the problem of correlation that can occur with the Halton sequence for higher dimensions. For example, consider the base of the sequences for 100 dimensions. The last Halton sequence for a 100-dimension problem uses the 100th prime number: 541. The Faure sequence uses the smallest prime that is larger than or equal to 100: 101.

Hammersley sequence: In general, the n-dimensional Hammersley set of size m contains a first component of successive fractions ${}^{0}/_{m}, {}^{1}/_{m}, \dots, {}^{m}/_{m}$paired with $n-1$ 1-dimensional van der Corput sequences, where the bases are the first $n-1$ primes. (Hammersley, 1960)

Sobol sequence: The Sobol sequence is described in (Antonov, 1979). The Sobol sequence has the same base for all dimensions and permutes the sequence of elements within each dimension. The Sobol sequence uses the coefficients of irreducible primitive polynomials with modulo 2. It is valid up to 1111 dimensions.

Niederreiter sequence: The generalized Niederreiter sequence is described in (Niederreiter, 1988). Its construction is based on the theory of (t, s)-sequences. (Niederreiter, 1987) It is valid up to 20 dimensions.

The raw result of an uncertainty quantification using any of the sampling strategies described above is a median with confidence interval centered about it. During the simulation setup, the user is asked for a confidence level larger than 0.5 and smaller than 1.0. Cumulative distribution function (CDF) uncertainty quantification involves a family of CDFs - one for each sample. The user can opt to have the raw data post-processed and output the median failure function with its confidence limits defined by the confidence level (Figure 4-5).

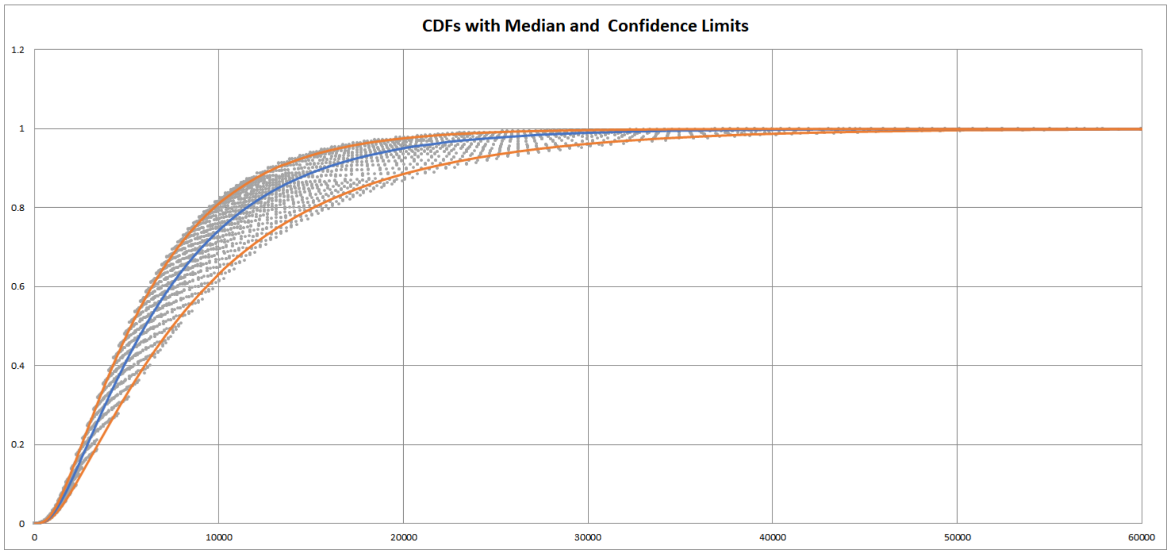


Figure 4-5: Family of failure functions (gray) with overlapping median (blue) and confidence limits (orange)

4.3 Importance measures

Importance measures are key ingredients of PRA used to rank the relative contributions to risk between end states or components in a system.

4.3.1 Conditional importance measure

The conditional importance measure of a component is based on the time-dependent conditional probability of system failure given that component has already failed:

$$I^{conditional}(i|t) = P_s(F_s|F_i, t)$$

4.3.2 Marginal importance measure

The marginal or Birnbaum's importance measure quantifies the rate of change of the system reliability because of changes to the reliability of a single component. Its mathematical expression is:

$$I^{marginal}(i|t) = \frac{\partial R_s(t)}{\partial R_i(t)}$$

If the importance measure is large for a component i, then a small change in the reliability of component i results in a large change in the system reliability at time t. The marginal importance measure can be interpreted to be the probability that at time t component i is critical for the system.

Also note that the marginal importance measure of component i is independent of the actual reliability of component i. In other words, it only depends on the structure of the system and the reliabilities of the other components.

In practice, components with a very low value of the marginal importance measure have a small effect on the system reliability, thus requirements for finding highly reliability components to perform their functions may be relaxed. On the other hand, components with a very high value of the marginal importance measure are critical for the system at that time t; therefore, a lot more effort should be put into finding components with higher reliability, finding higher quality reliability data, or even changing the structure of the system.

4.3.3 Improvement potential

The improvement potential importance measure is the difference between the system reliability with an ideal component i (that is, its reliability is equal to 1), and the system reliability with the actual component i. Mathematically, this is defined as:

$$I^{potential}(i|t) = R(t|1_i) - R(t)$$

The improvement potential importance measure, as its name is suggesting, indicates how much it is possible to improve the current system reliability by replacing component i with an ideal component.

4.3.4 Criticality

The criticality importance measure is defined as the probability that component i is critical for the system and is failed at time t, given that the system is failed at time t. It can be obtained from the marginal importance measure in the following way:

$$I^{criticality}(i|t) = \frac{I^{marginal}(i|t) \cdot p_i(t)}{p_s(t)}$$

In other words, the criticality importance measure gives a measure of the probability that a component i causes the system to fail. Therefore, if the component i is repaired, the system is expected to function again. The prioritizing of maintenance or repair actions in complex systems can be accomplished with the criticality importance measure.

4.3.5 Diagnostic importance measure

The diagnostic or Fussell-Vesely importance measure is the probability that at least one minimal cut set that contains component i is failed at time t, given that the system is failed at time t. It is expressed as:

$$I^{diagnostic}(i|t) = \frac{\sum_{j=1}^{mcs_i} p_j^i(t)}{p_s(t)}$$

In practice, it should give similar results as the criticality importance measure, thus they can be used for the same scope.

4.3.6 Risk achievement worth

The risk achievement worth (RAW) importance measure quantifies the relative increase in the system failure given that component i is in a failed state:

$$I^{RAW}(i|t) = \frac{F_s(t|1_i)}{F_s(t)}$$

In practice, the RAW importance measure is used to find the risk significance of components that are removed from the system. If the importance measure is close to 1, then its improvement has negligible effect on the system.

4.3.7 Risk reduction worth

The risk reduction worth (RRW) importance measure quantifies the relative reduction in the system failure given that component i is functioning:

$$I^{RRW}(i|t) = \frac{F_s(t)}{F_s(t|0_i)}$$

The basic event i may sometimes represent an operator action instead of a component failure. For such cases, it may be useful to analyze the effect of operator inaction on the mission success. It is like to the critically importance measure.

4.4 Leaky noisy OR gates for Bayesian networks quantification

Canonical probabilistic nodes, such as noisy OR gate, are convenient knowledge engineering tools widely used in practical applications. The noisy OR gates were introduced for binary variables by Pearl (1988) and extended to binary leaky noisy OR gates by Henrion (1989). The noisy OR often approximates the true distribution of the conditional probabilities while also significantly reducing the effort required in building the conditional probability table. In case of a general BBN binary node with n binary parents, the number of probabilities is 2^n. This number can quickly become prohibitive (for example, 10 nodes results in a conditional probability table that contains 2^{10} or

1024 probabilities). A noisy OR model needs only n+1 probabilities, that is the number of probabilities required for completing the conditional probability table grows linearly rather than exponentially as the number of parents increase.

The leaky noisy OR gates are an extension of the noisy OR gates that include an extra parameter called the leak factor. The leak factor is the probability that the events are true when all their causes are absent.

The Structural Modeling, Inference, and Learning Engine library[2] was used to implement an algorithm to quantify the BBNs containing leaky noisy OR gates in the HCL library.

[2] https://www.bayesfusion.com/smile-engine

5 Introduction of dynamically linked FTs modeling

As described in Chapter 1, the hardware reliability model allowed the modeling of two types of hardware failure events: time dependent failures, and conditional demand failures. Time dependent failures allow the analyst to initiate hardware state changes, including failures, at a prescribed time during the simulation. Conditional or on demand failures are triggered when a specified component changes its operating state. These types of failures fall into the category of frontline system failures. The ADS-IDAC hardware reliability model was limited in its ability to capture support system failures and their impact on the frontline systems entirely. Common examples of support systems are service water, component cooling water, instrument or plant air, or backup power. Their failure can cause a reactor trip or affect the frontline systems necessary to mitigate a possible NPP transient. The main challenge in modeling the support systems is that they cannot be modeled with the thermal-hydraulic RELAP5 code.

FT analysis is one of the most commonly used ingredients of PRA with many models readily available in the literature or databases. Acknowledging this, a dynamic extension of the FT analysis was adopted in ADS-IDAC. The frontline and support system failure capability was implemented with the aid of dynamically linked FTs to the system module, hardware reliability module, and scheduler for creating DDETs branching points.

The benefits for employing such an approach is that the failure events are moved from the system level to the component level. In turn, this allows for better capturing the importance of components in the frontline and support systems, plant-to-plant variability can be more appropriately simulated, and when these systems are needed their success probability is better estimated using the actual state of the components included in the dynamic FT. Moreover, partial frontline system failures that depend on the status of the support systems can be modeled, for example the availability of one train of component cooling water versus the availability of two trains of component cooling water. Moreover, the component cooling water pumps need power to operate are automatically loaded on the emergency diesel generator in the event of a loss of AC power. To simulate such a scenario, FTs for the emergency diesel generators can be connected to the component cooling water system FT through common transfer gates.

The most efficient way to incorporate frontline and support systems failure logic and reliability into ADS-IDAC was to utilize the HCL library described in Chapter 2.2. The computational architecture compatibility between the C++ HCL code and C++/Fortran ADS-IDAC allowed the HCL to be internally coupled to ADS-IDAC through an interface as a static library (Figure 5-1).

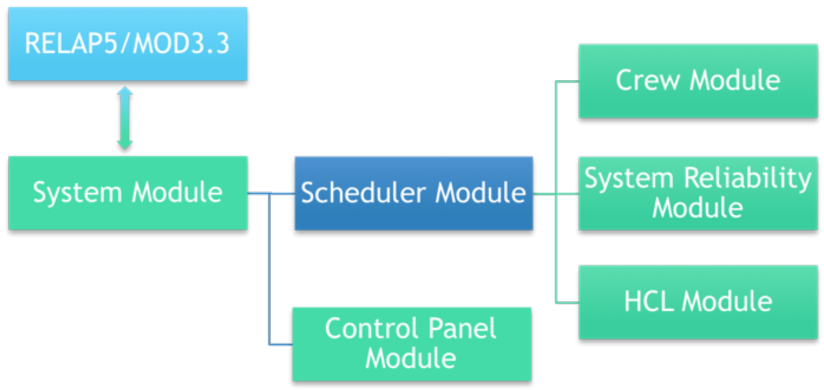


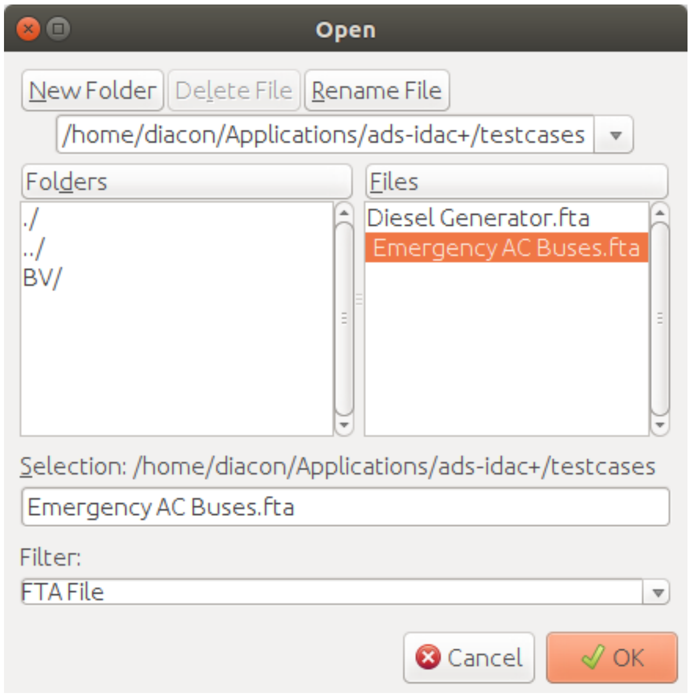
Figure 5-1 Updated ADS-IDAC configuration with HCL module

An advantage in choosing the HCL library is the fact that BBNs are an integral part of HCL, making it appropriate to realistically capture the soft causal dependencies such as environmental exposure, organizational factors, or maintenance practices. The South Texas Nuclear Project Unit 1 the auxiliary feedwater pumps had been susceptible to stress corrosion cracking in their bushings because the shaft sleeves were not made of the appropriate stainless streel material. All four auxiliary feedwater pumps were affected. However, only one failed during a performance test in 1988. The pump shaft of the affected pump failed during stress loading while it was running. It was discovered that all the auxiliary feedwater pumps were affected by the same common design deficiency. This type of common cause failure can be appropriately modeled through a BBN for the degradation mechanism and connecting it to all shaft sleeves of the auxiliary feedwater pumps. In this way, the dependencies between the same components used for redundancy and diversity are explicitly included and the system failure probability is appropriately estimated.

Internal ADS-IDAC updates were necessary to be implemented for accommodating support systems and frontline systems modeled with an FT through the top event and basic event nodes for the following types of C++ objects:

- Updates in the control panel module give access to the existing systems and operators to the status of those components and systems modeled with dynamic FTs. In turn, through the control panel, the operational status of frontline systems modeled with dynamic FTs is made available to the system module and subsequently to the RELAP5 code to affect the thermal-hydraulic system behavior.
- Changes were made to the scheduler module to include new branching rules that include failures of frontline and supports systems.
- The hardware reliability module was updated to include the failures of frontline and support systems on demand, during operation, and time dependent initiating events.
- Update to the crew module were performed to include frontline and support systems in procedures, heuristic diagnosis, and reasoning machine.

Moreover, the internal ADS-IDAC changes required corresponding updates to the input file requirements, in addition to the input files that define the FTs and BBNs desired to be modeled into the simulation. The FTs and BBNs input files use the XML format defined in the HCL algorithm. The user can also employ the IRIS graphical user interface, to create the FTs and BBNs that are imported into ADS-IDAC (Figure 5-2).


Selection: /home/diacon/Applications/ads-idac+/testcases

Emergency AC Buses.fta

Filter: FTA File

Figure 5-2 Importing of '.fta' FT XML formatted files into ADS-IDAC graphical user interface

The typical techniques of constructing an FT can be readily applied: system decomposition, functional analysis, and, eventually, defining the top event and successive subordinate failure events connected through logical gates. The system or subsystems can be decomposed up to the basic events. All basic events that can be assumed as statistically independent are assigned a probability or failure distribution. However, if they are identified as common cause failures, they can be linked to a BBN for further modeling and quantification.

The graphical representation of a dynamically linked FT into the generation of a DDET is illustrated in Figure 5-4 showing a possible system evolution. Using this hypothetical example, the implemented algorithm for modeling the impact of support systems on the frontline systems is described for the highlighted sequence in red.

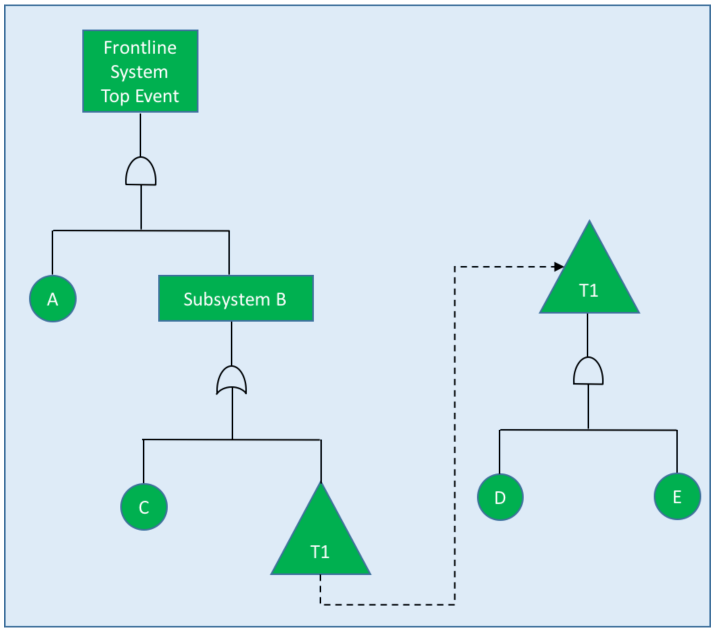


Figure 5-3 Frontline system with support system connected through a transfer gate T1

Three snapshots are overlaid on top of the DDET to graphically show the status of the frontline and support system components during this sequence. Two FTs, one for the frontline system (i.e., Frontline System) and one for the support system, are linked through a transfer gate, T1. (Figure 5-3) The green colored nodes indicate operational availability, while the red color node indicates failure. The frontline system is made of component A and subsystem B logically connected through an AND gate. Subsystem B requires both component C and support system T1 to successfully function. Support system T1 is a parallel system made of two components D and E. Therefore, the Frontline system fault expression when all the components are available is:

$$F_{Frontline\ System}\ (@t = 0) = A \cdot C + A \cdot D \cdot E$$

The red sequence end state probability is calculated by multiplying the initiating event, all the branching events, and the end state in the following way:

$$ES\ \equiv I \cdot PE_1 \cdot PE_2 \cdot \overline{PE_3} \cdot \overline{PE_4} \cdot PE_5 \cdot \overline{PE_6} \cdot PE_7 \cdot \overline{PE_8} \cdot \overline{PE_9}$$

Where I is the initiating event, PE$_i$ are branching events, and ES is the end state. Hence, by including the failures and the fault expression of the dynamically linked FTs at various time steps into the end state, the end state expression is written in the following way:

$$ES \equiv I \cdot PE_1 \cdot PE_2 \cdot A \cdot F_{Frontline\ System}\left(@t = t_{PE_4}\right) \cdot PE_5 \cdot D \cdot$$

$$\cdot F_{Frontline\ System}\left(@t = t_{PE_7}\right) \cdot \overline{PE_8} \cdot E$$

where

$$F_{Frontline\ System}\left(@t = t_{PE_4}\right) = C + D \cdot E$$

and

$$F_{Frontline\ System}\left(@t = t_{PE_7}\right) = C + E$$

A few events after the initiating event is the failure of component A of the Frontline System. At this time, the Frontline System automatically switches its demand to the Subsystem B that successfully starts; therefore, when it is needed for a while, the Frontline system is able to start, but its probability of failure may be significantly higher later during the simulation. After the Frontline system is switched to standby, component D of the support system T1 fails. Note that due to the functional logic of the systems, the Frontline System is still functional. Later, the failure of Component E makes the support system T1 unavailable, and subsequently the Frontline System fails. Nonetheless, when the Frontline system is needed again, it successfully starts to operate.

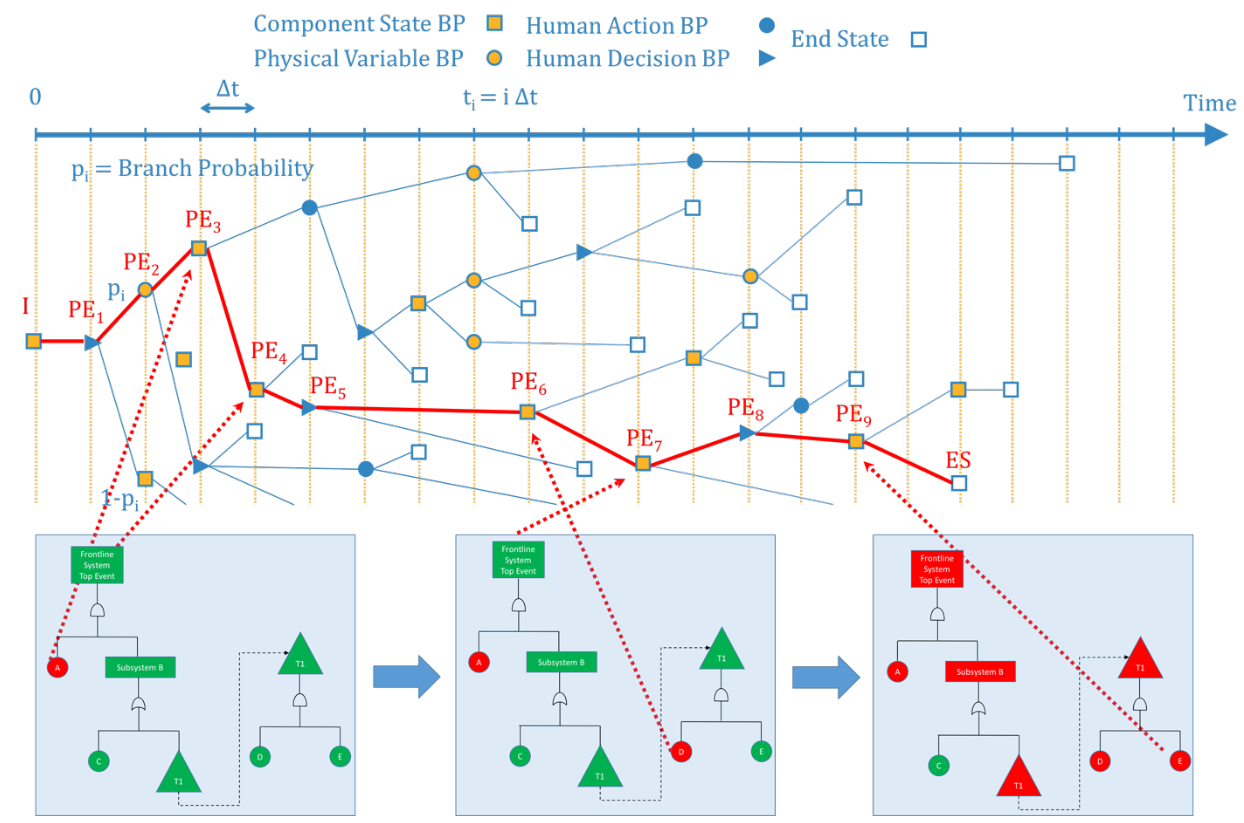


Figure 5-4. Graphical representation of dynamically linked FTs into the generation of a DDET.

Finally, because the failure of component E disables the Support System T1, it implicitly fails the Frontline System and the sequence ends because failure of the Frontline System leads to core damage. In Chapter 9, a real accident scenario is simulated to further showcase the modeling of the support systems and their impact on the frontline systems through dynamic FTs.

The ADS-IDAC hardware reliability module was updated to include the modeling of support system failures at fixed time and on demand similarly to previously supported frontline system failures. Moreover, ADS-IDAC has been extended to cover the modeling of both frontline and support system failures during operation by considering the failure rate, the number of failures selected and the time interval between them. This feature further extends ADS-IDAC's capability to dynamically predict the timing importance of component failures during operation for the overall safety of the design in question.

6 New quantification model for human error

In this chapter, a HFE quantification framework that is implemented into ADS-IDAC is described. As the frontline and support system failures are already modeled in the hardware reliability module, this allows ASD-IDAC to fully quantify the generated DDET.

6.1 IDAC human behavior adjusted by context and operator variability

During the simulation, the human operator behavior in IDAC is adjusted based on the context through a mechanism of surrogates – PSFs – manifestation nodes (Figure 6-1). At each time step, the NPP state parameters are used to adjust the surrogate node values, the surrogates (yellow nodes) affect dependent PSFs (blue nodes) and in turn the PSFs affect manifestation (green nodes). The relationships between these nodes are based on empirical correlations found through extensive literature reviews corresponding to the appropriate human behavior mechanisms (Coyne, 2009) (Li, 2013).

Like all information processed by the operator model, all the dynamic PSF values are based on information perceived by the operator rather than data obtained directly from the thermal-hydraulic model or control panel. Perceived data may differ from the actual parameter value in thermal-hydraulic model or control panel due to time lags in updating perceived data and any distortions introduced by perception filtering and biasing. All the PSFs modeled in ADS-IDAC are briefly described below:

- System and Parameter Criticality [Time constraint]: The criticality of system condition dynamic PSF represents the operator's perception of the level of degradation of key safety functions compared to normal operation. This PSF is based on the safety parameter display system used in NPP control rooms. The value of the system criticality PSF corresponds to the aggregate deviation of key safety parameters from a nominal value. Each operator profile has its own parameters used to calculate this PSF: the threshold limits associated with each parameter, and the weighting factors used to aggregate the parameter contributions. Typical parameters used to calculate the system criticality PSF include RCS subcooling margin, wide range steam generator water levels, pressurizer water level, and reactor vessel water level. The contribution from each identified parameter to the overall criticality of system condition PSF value is denoted as the parameter criticality. Given a set of high and low threshold limits, the parameter criticality corresponds to the magnitude of the parameter's deviation from a nominal safe condition.

- Information Load [Resources]: The information loading dynamic PSF represents the operator's mental workload associated with the perception, processing, and communication of information. All information available from the NPP hydraulic model and crew communications must first pass through the operator's perception filter before it can be memorized and used. Consequently, the information flow rate through the perception filter provides an appropriate measure of each operator's information processing workload.
- Time Constraint Load [Time constraint]: The time constraint load dynamic PSF represents the time available until a monitored NPP parameter exceeds a critical threshold. Because operators will normally monitor more than one important parameter, the overall PSF value is based on the most time critical parameter. The knowledge base profile for each operator includes data defining how the time constraint load PSF value is calculated, including a listing of NPP parameters used to calculate the time constraint PSF value along with the associated critical threshold values. Typical parameters that may be included in the

calculation of the time constraint PSF include steam generator water levels, pressurizer water level, and RCS pressure.

- Cognitive Task Load [Task load]: The task load dynamic PSF is indicative of the actual task demand assigned to a person quantified in terms of the number and type of tasks in a time unit. NPP control room operations do not normally involve heavy physical work, so in ADS-IDAC only the cognitive task load is of interest. Simulation HRA models possess a unique advantage of tracking each activity performed by the operator, which allows the code to count and to assess the workload specifically; therefore, the cognitive task load is also a dynamic PSF evaluated at each time step.
- Passive Alarm Load [HSI]: The passive alarm load dynamic performance factor embodies the number of salient stimuli that catch the operator's attention automatically like the alarms in the control room. Most often passive information is intrusive and grabs the operators' attention while interrupting their ongoing cognitive processes. Thus, too much passive information could be overwhelming. In addition to causing mental stress, it shifts one's attention and impedes the ability to refocus.
- Expertise [Knowledge/Abilities]: Operator expertise facilitates operator's coping with fast system dynamics in several ways: structuring and sorting the observations systematically, speeding the retrieval of knowledge for explaining the observation, and making connections between different pieces of information.

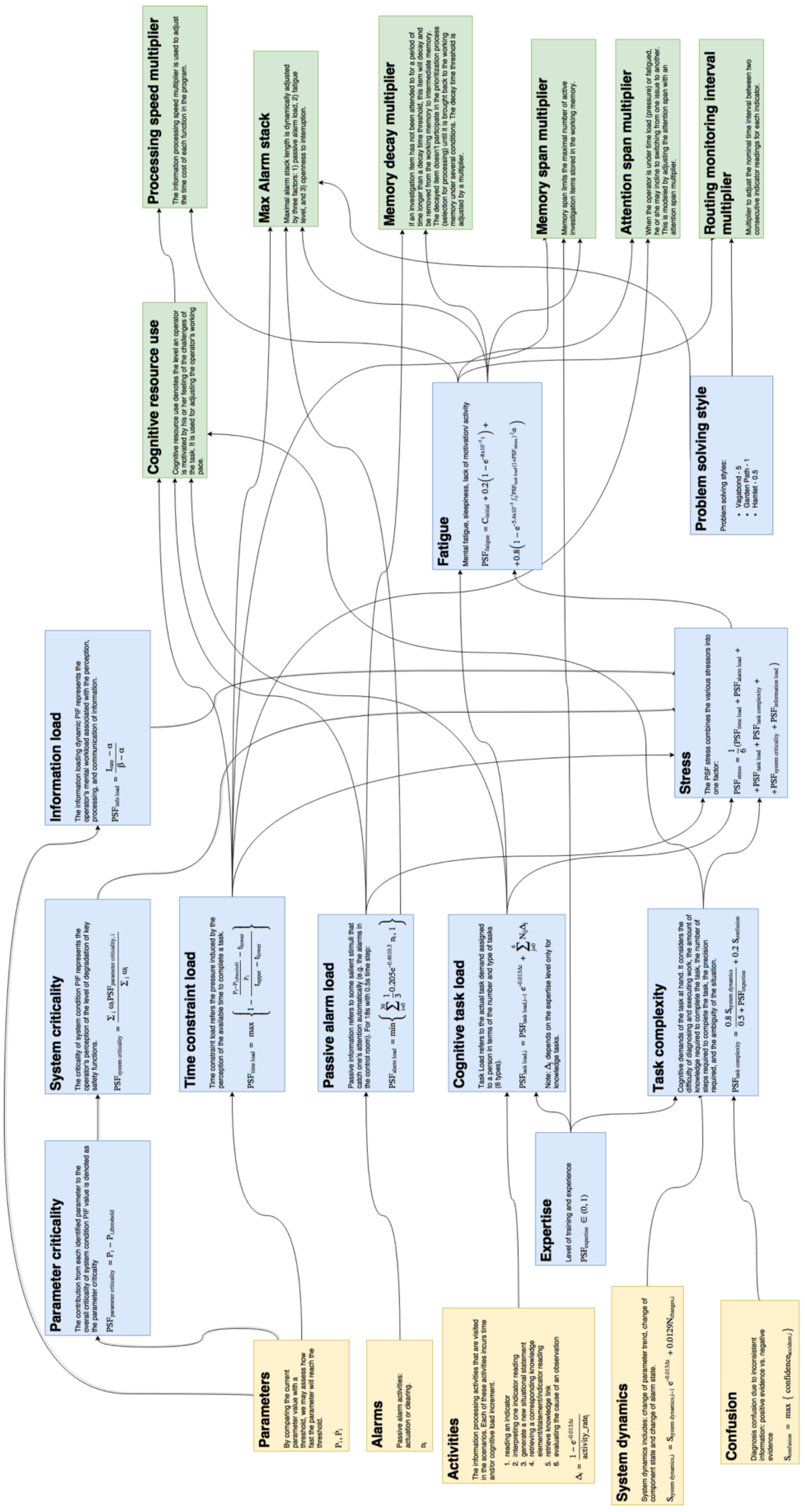

Parameters
By comparing the current parameter value with a threshold, we may assess how fast the parameter will reach the threshold.
P_i, P_t
Alarms
Passive alarm activities: actuation or clearing.
a_t
Activities
The information processing activities that are visited in the scenarios. Each of these activities incurs time and/or cognitive load increment.
1. reading an indicator
2. interpreting one indicator reading
3. generate a new situational statement
4. retrieving a corresponding knowledge element/statement/indicator reading
5. retrieve knowledge link
6. evaluating the cause of an observation
$A_i = \frac{1-e^{-0.013t_i}}{activity\_rate_i}$
System dynamics
System dynamics includes: change of parameter trend, change of component state and change of alarm state.
$S_{system\ dynamics,i} = S_{system\ dynamics,i-1} \cdot e^{-0.013\Delta t} + 0.0129N_{changes,i}$
Confusion
Diagnosis confusion due to inconsistent information: positive evidence vs. negative evidence
$S_{confusion} = \max\{confidence_{incident,i}\}$
Parameter criticality
The contribution from each identified parameter to the overall criticality of system condition PIF value is denoted as the parameter criticality
$PSF_{parameter\ criticality} = P_i - P_{threshold}$
System criticality
The criticality of system condition PIF represents the operator's perception of the level of degradation of key safety functions.
$PSF_{system\ criticality} = \frac{\sum_i \omega_i PSF_{parameter\ criticality,i}}{\sum_i \omega_i}$
Information load
The information loading dynamic PIF represents the operator's mental workload associated with the perception, processing, and communication of information.
$PSF_{info\ load} = \frac{I_{max}-\alpha}{\beta-\alpha}$
Time constraint load
Time constraint load refers the pressure induced by the perception of the available time to complete a task.
$PSF_{time\ load} = \max\left\{1-\frac{\frac{P_i-P_{threshold}}{P_i}-t_{lower}}{t_{upper}-t_{lower}}\right\}$
Passive alarm load
Passive information refers to some salient stimuli that catch one's attention automatically (e.g. the alarms in the control room). For 18s with 0.5s time step:
$PSF_{alarm\ load} = \min\left\{\sum_{i=0}^{36}\frac{1}{3}0.205e^{-0.461i0.5}a_i, 1\right\}$
Cognitive task load
Task Load refers to the actual task demand assigned to a person in terms of the number and type of tasks (6 types).
$PSF_{task\ load,i} = PSF_{task\ load,i-1} \cdot e^{-0.013\Delta t} + \sum_{j=0}^{6} N_j A_j$
Note: A_j depends on the expertise level only for knowledge tasks.
Expertise
Level of training and experience
$PSF_{expertise} \in (0,1)$
Task complexity
Cognitive demands of the task at hand. It considers the difficulty of diagnosing and executing work, the amount of knowledge required to complete the task, the number of steps required to complete the task, the precision required, and the ambiguity of the situation.
$PSF_{task\ complexity} = \frac{0.8\ S_{system\ dynamics}}{0.5 + PSF_{expertise}} + 0.2\ S_{confusion}$
Stress
The PSF stress combines the various stressors into one factor:
$PSF_{stress} = \frac{1}{6}(PSF_{time\ load} + PSF_{alarm\ load} + PSF_{task\ load} + PSF_{task\ complexity} + PSF_{system\ criticality} + PSF_{information\ load})$
Fatigue
Mental fatigue, sleepiness, lack of motivation/ activity
$PSF_{fatigue} = C_{initial} + 0.2\left(1-e^{-8.6\cdot10^{-5}t}\right) + 0.8\left(1-e^{-5.4\cdot10^{-5}\int_0^t PSF_{task\ load}(1+PSF_{stress})^2 dt}\right)$
Problem solving style
Problem solving styles:
Vagabond - 5
Garden Path - 1
Hamlet - 0.5
Cognitive resource use
Cognitive resource use denotes the level an operator is motivated by his or her feeling of the challenges of the task. It is used for adjusting the operator's working pace.
Processing speed multiplier
The information processing speed multiplier is used to adjust the time cost of each function in the program.
Max Alarm stack
Maximal alarm stack length is dynamically adjusted by three factors: 1) passive alarm load, 2) fatigue level, and 3) openness to interruption.
Memory decay multiplier
If an investigation item has not been attended to for a period of time longer than a decay time threshold, this item will decay and be removed from the working memory. The decayed item doesn't participate in the prioritization process (selection for processing) until it is brought back to the working memory under several conditions. The decay time threshold is adjusted by a multiplier.
Memory span multiplier
Memory span limits the maximal number of active investigation items stored in the working memory.
Attention span multiplier
When the operator is under time load (pressure) or fatigued, he or she may incline to switching from one issue to another. This is modeled by adjusting the attention span with an attention span multiplier.
Routing monitoring interval multiplier
Multiplier to adjust the nominal time interval between two consecutive indicator readings for each indicator.

Figure 6-1 Surrogates (yellow) – PSFs (blue) – Manifestation Nodes (green) IDAC model

- Task Complexity [Procedures]: The task complexity dynamic PSF represents a measure of interaction among system dynamics, diagnosis confusion, and operator expertise. Amongst the system dynamics tracked at each time step are parameter trend changes, component state changes, and alarm state changes. Diagnosis confusion represents the complexity induced by inconsistent information and indicates the operator's level of understanding of the current NPP status.
- Stress [Stress]: The stress dynamic PSF combines various stress inducing PSFs into one factor: time constraint load, passive information load, cognitive task load, and task complexity. Each of the stressors has an equal weight on the stress value.
- Fatigue [Stress]: As the NPP control room operators' tasks do not involve heavy physical work. Thus, only the following three dimensions are considered in calculating this dynamic PSF: mental fatigue, sleepiness, lack of motivation/ activity. It is evaluated based on an initial fatigue level at the beginning of their shift, a prolonged effort component due to performing tasks over a long period of time, and a sustained effort component representing the accumulation of fatigue by performing tasks. Moreover, the sustained effort component of fatigue is accelerated by the stress level.
- Problem-Solving Style [Team Effectiveness]: The problem-solving style static PSF is reflected into the following of model parameters and information processing functions. In ADS-IDAC three problem solving styles have been implemented: Vagabond, Hamlet, and Garden-Path styles. They affect various parameters used to model the variation in diagnosis of operators in the reasoning module: routine monitoring time interval, maximal alarm stack length, prioritization of investigation items, investigation termination criteria, and accident awareness thresholds.

This framework performs well for adjusting the behavior of human operators based on the context. However, it is incomplete as it does not include any HFE, crew failure mode, or HEP quantification that must be included in the DDET events for its full quantification. In the conventional HRA methods, like SPAR-H, the HEP is quantified by adjusting a nominal HEP dependent on the type of human event based on the value of a series of static PSFs. Based on this simple model, a dynamic framework can be envisioned to extend the surrogates – PSF –manifestation nodes found in ADS-IDAC like the one in Figure 6-2. In the figure, the red nodes are used to dynamically calculate the HEP at each time step for several activities performed by the operators like information gathering, diagnosis, procedure following, and action taking. As in the conventional SPAR-H quantification of the HEP, a nominal HEP for each type of event is adjusted based on the PSFs values. A composite PSF is used to give various weights to the PSFs.

This model would be a useful extension of the current HRA practices that leverages the unique advantages of a dynamic PRA platform of tracking the system and operator behavior. Nonetheless, it was not implemented into ADS-IDAC given its limitations in capturing the causal relationships between the PSFs and the types of human failure events (HFEs). A quantification framework based on a DBN model of the crew failure modes and PSFs was adopted inside ADS-IDAC for several reasons described in the next section.

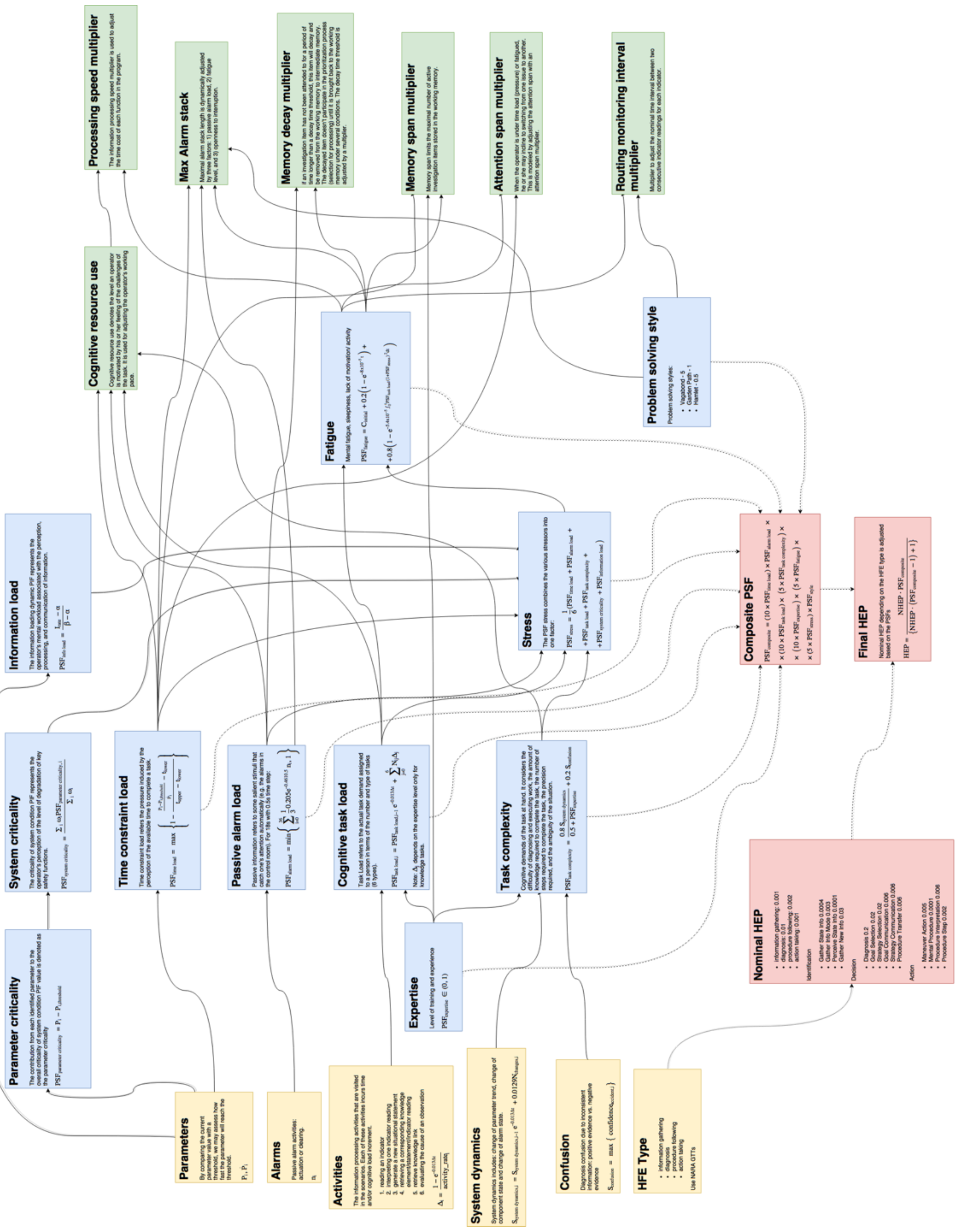


Figure 6-2 Extended Surrogates (yellow) – PSFs (blue) – Manifestation Nodes (green) IDAC model

6.2 Human failure events quantification through a DBN model

The starting point for developing the DBN model of the ADS-IDAC human failure event (HFE) probability quantification was the Phoenix method developed by Ekanem. It is a static HRA method that developed out of the IDAC model. It is a natural step to include the additional elements of the Phoenix method into the IDAC model as part of the full dynamic ADS-IDAC simulation environment.

The quantification through a BBN developed by Ekanem in the static HRA Phoenix method involves the construction of a BBN by using the crew failure modes and PSFs as nodes and the arcs to show the relationships of influence between them through a conditional probability table. BBNs provide numerous benefits such as the ability to incorporate both qualitative and quantitative information from different sources for analysis, a causal structure for modeling interdependencies among its elements, the flexibility of updating the present state of knowledge of the model to incorporate new evidence as it becomes available, the capability of reasoning under uncertainty, and its ability to interface with existing ET and FT PRA models.

A BBN is valuable for problem domains or systems where the variables are static (that is, they do not change over time). Static variables cannot always be assumed. For example, NPP system parameters and human operators' reasoning are clearly changing over time. In these cases, a DBN is necessary. A DBN is a BBN that is extended to incorporate a temporal dimension to enable the modeling of dynamic systems. The temporal extension of a BBN does not necessarily mean that the network structure or parameters changes dynamically, but it means that a dynamic system is being modeled. Hence, a DBN is a directed, acyclic graphical model of a stochastic process. It consists of time steps, with each time step containing its own variable values. The basic idea in a DBN is to specify how variables at time t influence variables at time t+1 and replicating the structure of a model for each time step (Figure 6-3). This concept of the DBN was used to model the dependencies between the HFEs by replicating the network structure to represent the dynamic system and ultimately estimate the conditional HEP at each time step (Figure 6-4). This structured, causal model integrated into ADS-IDAC also helps improve the reproducibility and transparency of results produced by different HRA analysts for the same scenario.

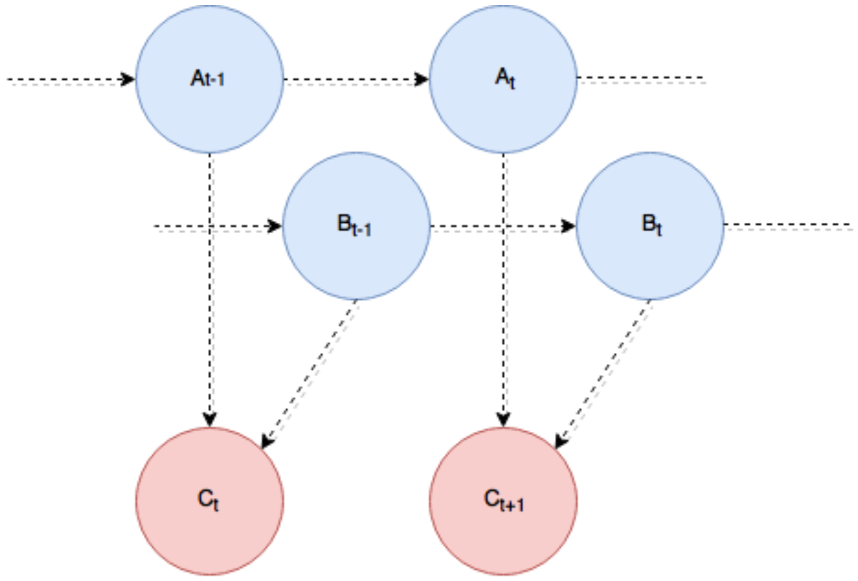


Figure 6-3 Simplified DBN with developed on one time step with two dynamic nodes A and B influencing node C at every time step

Construction of the DBN involves building the structure of the network and defining the data describing the causal relationships between the network's nodes, and the nodes that will change in time. The starting point in developing this DBN is the BBN developed in the Phoenix method. Unfortunately, this network cannot be adopted without modifications as some of its nodes do not have an ADS-IDAC equivalent and they do not cover all the HFEs modeled by ADS-IDAC.

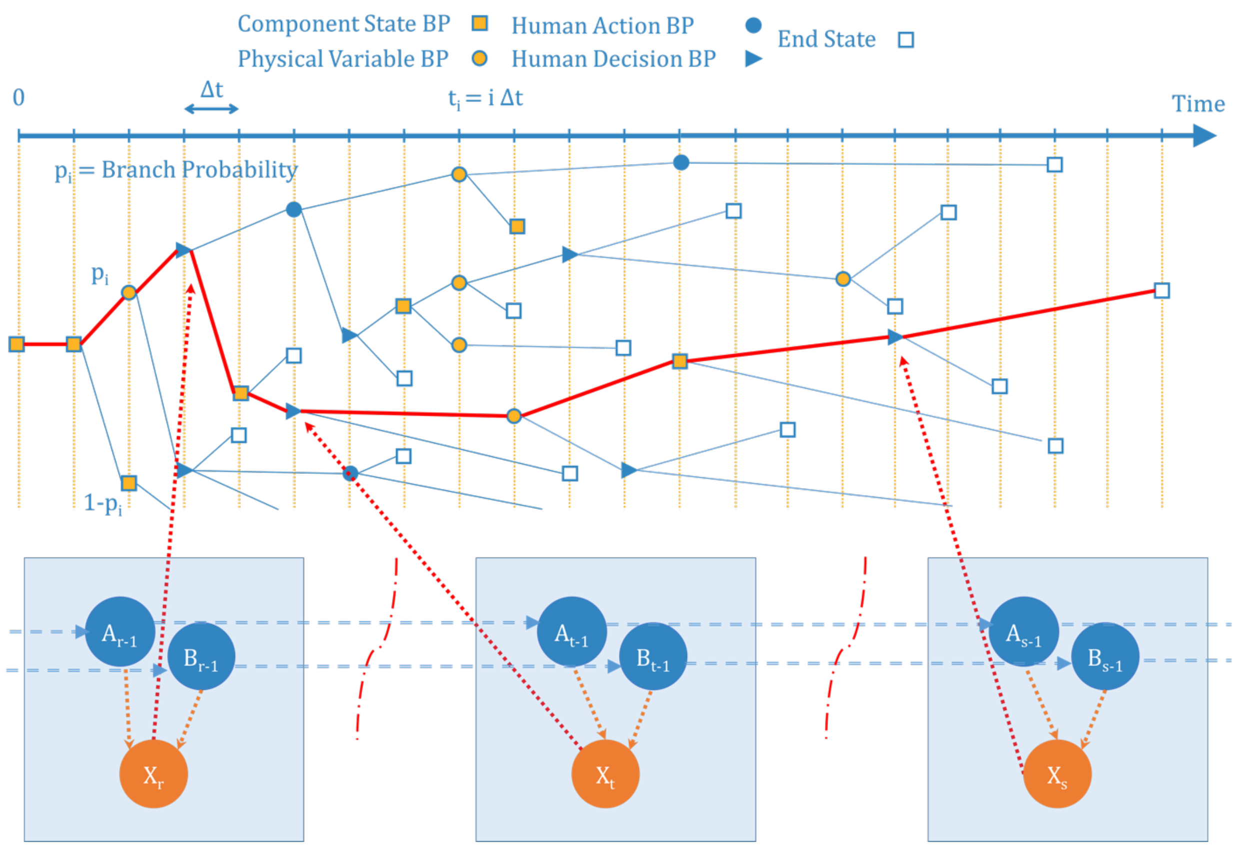


Figure 6-4 Graphical representation of an ADS-IDAC generated DDET where human events quantified with a simplified DBN are highlighted

The structure of the DBN contains two layers in which all the top layer nodes influence all the bottom layer nodes (Figure 6-5). Since the primary purpose of this DBN is to model the effect of the PSFs on the crew failure modes, the top layer contains the PSF and the bottom layer contains crew failure modes. The top layer contains the PSFs described in the previous section: system criticality, information load, time constraint load, cognitive task load, passive alarm load, expertise, task complexity, stress, fatigue, and problem-solving style. As the Phoenix method does not have the same PSFs, based on their definition and purpose an equivalence relationships table was created to match the PSF used in ADS-IDAC and Phoenix (Table 6-1). All the PSFs except the expertise and problem-solving style are dynamic; therefore, their value will change as the simulation progresses depending on the context. The PSFs have also been normalized to have values between 0 and 1.

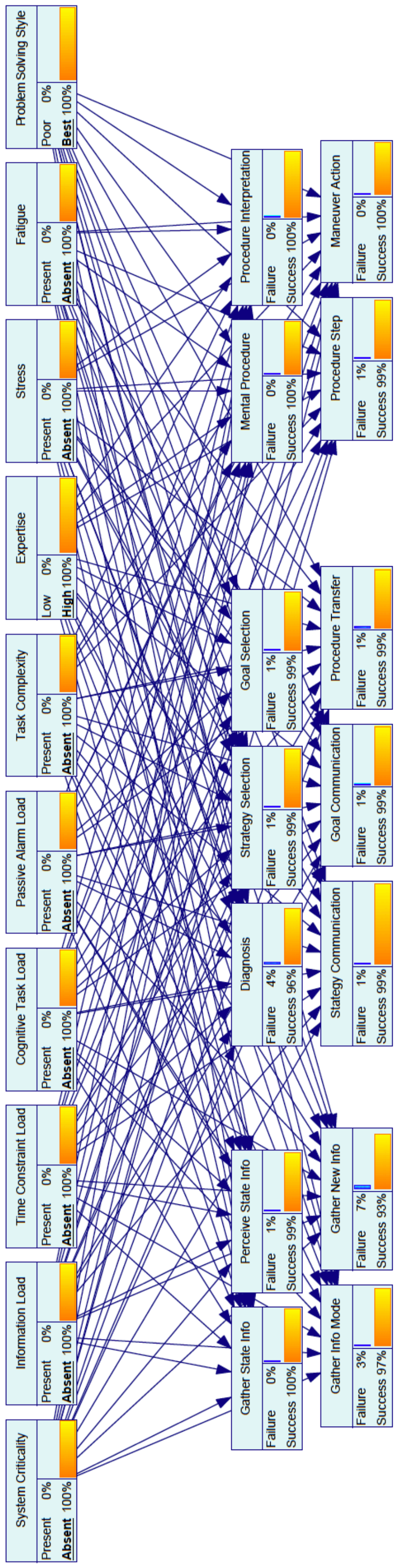


Figure 6-5 BBN of PSFs and HFEs

Table 6-1 Mapping between the PSFs of ADS-IDAC and Phoenix

ADS-IDAC	**Phoenix**
System criticality	Time constraint
Information load	Resources
Time constraint load	Time constraint
Cognitive task load	Task load
Passive alarm load	HSI
Expertise	Knowledge/ Abilities
Task complexity	Procedures
Stress	Stress
Fatigue	Stress
Problem-solving style	Team effectiveness

The bottom layer of the DBN in Figure 6-5 is made of crew failure modes. As in the Phoenix method, the crew failures modes specify the possible forms of human error in each of the information pre-processing, decision-making, and action execution phases (Figure 1-4). Note that it is assumed the reasoning machine is included in the decision-making phase although graphically the reasoning module is outside the decision-making box. The crew failure modes are also the generic functional modes of failure of the crew in its interactions with the NPP and represent the manifestation of the crew failure mechanisms and proximate causes of failure. They are selected to cover the various modes of crew response including procedure driven, knowledge driven, or a hybrid of both. To avoid double counting crew failure scenarios during the estimation of HEPs, the crew failure modes are defined as being mutually exclusive.

The crew failure modes within the information phase assume that the crew has failed in detecting, noticing and understanding the plant function they are supposed to be handling. Human failure in this phase can be divided into two major groups namely: failure to perceive passive information and failure to actively gather information. The crew failure mode that would occur during the perceiving of passive information is “Perceive State Info” – the crew fails to perceive the plant parameters or states from the control panel. The crew failure modes that would occur during the active gathering of information are: “Gather State Info” – the crew unintentionally try to collect the information from the wrong source, “Gather Info Mode” – the crew decide to use the old memorized information instead of collecting updated information, and “Gather New Info” – the crew failure in gathering new information. The equivalence table between the ADS-IDAC and Phoenix crew failure modes in the information phase is given in Table 6-2.

Table 6-2 Mapping between the crew failure modes in the information pre-processing phase of ADS-IDAC and Phoenix

ADS-IDAC Operator Activity	Phoenix Crew Failure Mode
Perceive State Info	Reading Error
Gather State Info	Wrong Data Source Attended To
Gather Info Mode	Decision to Stop Gathering Data
Gather New Info	Data Incorrectly Processed

The crew failure modes within the decision-making phase assume that there is failure in situation assessment, problem solving and decision-making given correct information pre-processing. Therefore, the assumption is made that the crew has detected, noticed and understood the plant functions they are supposed to be handling. However, they have failed to make a correct assessment of the plant condition, diagnose, decide and plan the adequate response needed to solve the problem at hand. Moreover, the decision-making operator has the responsibility to communicate the action-taking operators the appropriate strategy. Ultimately, failures in this phase result in implementing an incorrect recovery strategy, hence failing the required function. Therefore, the following crew failure modes have been included: “Diagnosis” – decision-maker reaches the wrong assessment of the plant, “Strategy Selection” – decision-maker takes the wrong strategy given the correct situational assessment, “Strategy Communication” – decision-maker fails to communicate the correct strategy selected to the action-taker, “Goal Selection” – decision-maker selects the wrong immediate goal given the correct situational assessment, “Goal Communication” – decision-maker fails to communicate the correct goal selected to the action-taker, and “Procedure Transfer” – decision-maker switches to the wrong procedure. The equivalence table between the ADS-IDAC and Phoenix crew failure modes in the decision-making phase is given in Table 6-3.

Table 6-3 Mapping between the crew failure modes in the decision-making phase of ADS-IDAC and Phoenix

ADS-IDAC Operator Activity	Phoenix Crew Failure Mode
Diagnosis	Plant/ System State Misdiagnosed
Strategy Selection	Inappropriate Strategy Chosen
Strategy Communication	Information Miscommunicated
Goal Selection	Inappropriate Strategy Chosen
Goal Communication	Information Miscommunicated
Procedure Transfer	Inappropriate Transfer to a Different Procedure

The crew failure modes within the action execution phase involve failure in action execution given correct information pre-processing, situational assessment, and decision-making. It is assumed that the crew has detected, noticed and understood the NPP function they are supposed to be handling.

Also, it is assumed they have made a correct assessment of the NPP condition, diagnosed, decided and planned the adequate response needed to solve the problem. However, they fail in executing the response or required action. It is assumed that the crew failure modes in the action execution phase are unintentional errors, that is the operators are always acting in the interest of recovering the NPP. The following crew failure modes have been included: “Mental Procedure” where the crew fails to adapt the instinctive response procedure to the current situation, “Procedure Step” where the crew skip or pause a procedure step in order to rely of their knowledge, “Procedure Interpretation” where the crew misinterpret the procedure step expectation, and “Maneuver Action” where the action-taker does not perform the requested action. (Table 6-4)

Table 6-4 Mapping between the crew failure modes in the action execution phase of ADS-IDAC and Phoenix

ADS-IDAC Operator Activity	Phoenix Crew Failure Mode
Mental Procedure	Failure to Adapt Procedures to the Situation
Procedure Step	Procedure Step Omitted (Intentional)
Procedure Interpretation	Procedure Misinterpreted
Maneuver Action	Incorrect Operation on Component/ Object

Some of the crew failure modes fall into the category of errors of commission, that is they are the result of their intent given the wrong situational assessment of the NPP conditions.

Even if the structure of the DBN was defined, it still must to be integrated into ADS-IDAC by linking the all the human events types simulated into ADS-IDAC to the appropriate crew failure modes. For these human events, only success branches are generated and quantified by ADS-IDAC. Failure branches are covered in the next chapter.

The "Perceive State Info" crew failure mode is used to estimate the probability of the action-taker to correctly register the perceived information for an alarm state, a frontline system state, a support system state, a parameter value, and a parameter trend value from the control panel.

Table 6-5 Implementation locations of quantified "Perceive State Info" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorAT::infoChunking()	BranchExpectedAlarmState
OperatorAT::infoChunking()	BranchExpectedComponentState
OperatorAT::infoChunking()	BranchExpectedSupportSystemState
OperatorAT::infoChunking()	BranchExpectedParameterValue
OperatorAT::infoChunking()	BranchExpectedParameterDifference

The "Gather State Info" crew failure mode is used to estimate the probability of any of the human operators to collect the information from the correct source on the control panel: alarm state, frontline system state, support system state, or parameter value.

Table 6-6 Implementation locations of quantified "Gather State Info" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
Operator::ActiveFetchAlarmStateInfo()	BranchExpectedAlarmState
Operator::ActiveFetchComponentStateInfo()	BranchExpectedComponentState
Operator::ActiveFetchParameterValueInfo()	BranchExpectedParameterValue
Operator::ActiveFetchSupportSystemStateInfo()	BranchExpectedSupportSystemValue

The "Gather Info Mode" crew failure mode is used to estimate the probability any of the crew members succeeds in collecting updated information instead of using old memorized information.

In the previous version of ASD-IDAC, this value was fixed by the user at the beginning of the simulation and it remained unchanged.

Table 6-7 Implementation locations of quantified "Gather Info Mode" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
Operator::generateInfoGatherModeBranch()	BranchInfoGatherMode

The "Gather New Info" crew failure mode is used to estimate the action-taker's or decision-maker's probability of adding a parameter to the scan queue for gathering updated information.

Table 6-8 Implementation locations of quantified "Gather New Info" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorAT::oneThroughProcessForBlockI()	BranchScannedParameter
OperatorDM::oneThroughProcessForBlockI()	BranchScannedParameter

The "Diagnosis" crew failure mode is used to estimate the decision-maker's probability of reaching the correct assessment of the NPP given their understanding of the NPP conditions. Note that if the operators do not correctly understand the NPP conditions, they will still reach a diagnosis, even if it's the wrong one.

Table 6-9 Implementation locations of quantified "Diagnosis" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
DiagnosisActionModule::onceThroughProcessForReasoningStrategy()	BranchAction

The "Strategy Selection" crew failure mode informs the decision-maker's probability of selecting the appropriate strategy given the correct situational assessment. The supported strategy selections in ADS-IDAC are wait and monitor, procedure following, hardwired diagnosis, and knowledge-based reasoning.

Table 6-10 Implementation locations of quantified "Strategy Selection" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorDM::selectStrategy()	BranchStrategy

The "Strategy Communication" crew failure mode is used to estimate the decision-maker's probability of communicating to the action-taker the selected strategy. Moreover, if the action-taker is in the follow instruction strategy mode, the same crew failure mode is used to estimate the decision-maker's probability of communicating to the action-taker the appropriate instruction. The type of instruction can be to obtain information about an alarm state, a frontline system state, a support system state, a parameter value, and a parameter trend value from the control panel or change their values.

Table 6-11 Implementation locations of quantified "Strategy Communication" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorAT::selectStrategy()	BranchStrategy
OperatorAT::performFollowInstructionStrategy()	BranchAction
OperatorAT::performFollowInstructionStrategy()	BranchExpectedAlarmState
OperatorAT::performFollowInstructionStrategy()	BranchExpectedComponentState
OperatorAT::performFollowInstructionStrategy()	BranchExpectedSupportSystemState
OperatorAT::performFollowInstructionStrategy()	BranchExpectedParameterValue
OperatorAT::performFollowInstructionStrategy()	BranchExpectedParameterDifference

The “Goal Selection” crew failure mode is used to estimate the decision-maker’s probability of selects the appropriate immediate goal given the correct situational assessment. Selecting the inappropriate goals can lead a delay in the appropriate recovery actions.

Table 6-12 Implementation locations of quantified “Goal Selection” crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorDM::selectGoal()	BranchGoal

The “Goal Communication” crew failure mode is used to estimate the decision-maker’s probability to communicate the correct goal selected to the action-taker and consultant.

Table 6-13 Implementation locations of quantified “Goal Communication” crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorAT::selectGoal()	BranchGoal
OperatorCT::selectGoal()	BranchGoal

The “Procedure Transfer” crew failure mode is used to estimate the crew’s probability to switch to the correct written or mental procedure.

Table 6-14 Implementation locations of quantified “Procedure Transfer” crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
ProcedureStep::executeInstinctiveResponseProcedureTransfer()	BranchProcedureStep
ProcedureStep::oneThroughProcessForProcedureStep()	BranchProcedureStep
ProcedureStepUnit::executeAction()	BranchProcedureStep

The “Mental Procedure” crew failure mode is used to estimate the crew’s ability to perform the appropriate instinctive response procedure based on the correct situational assessment.

Table 6-15 Implementation locations of quantified “Mental Procedure” crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
Operator::performAMentalProcedure()	BranchProcedureStep
ExpectationORUnit::assessExpectation()	BranchExpectedMentalBelief
Operator::checkAndActivateHWKB()	BranchExpectedMentalBelief

The “Procedure Step” crew failure mode is used to estimate the crew’s probability of correctly skipping or pausing a procedure step in order to rely of their knowledge.

Table 6-16 Implementation locations of quantified “Procedure Step” crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
OperatorDM::performFollowProcedureStrategy()	BranchProcedureStep
OperatorDM::performFollowProcedureStrategy()	BranchProcedureStep
OperatorDM::performRecoveryAction()	BranchProcedureStep

The “Procedure Interpretation” crew failure mode informs the crew’s probability to correctly interpret the procedure step expectation. As in the case of perceiving information, the expectations can be related to an alarm state, a frontline system state, a support system state, a parameter value, or a parameter trend value from the control panel.

Table 6-17 Implementation locations of quantified "Procedure Interpretation" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
ExpectationORUnit::assessExpectation()	BranchExpectedAlarmState
ExpectationORUnit::assessExpectation()	BranchExpectedComponentState
ExpectationORUnit::assessExpectation()	BranchExpectedSupportSystemState
ExpectationORUnit::assessExpectation()	BranchExpectedParameterValue
ExpectationORUnit::assessExpectation()	BranchExpectedParameterDifference

The "Maneuver Action" crew failure mode is used to estimate the action-taker's probability to complete an action communicated by the decision-maker or from procedures.

Table 6-18 Implementation locations of quantified "Maneuver Action" crew failure mode in ADS-IDAC

ADS-IDAC Object::Function	Event Created
ManeuverActionUnit::executeManeuverControl()	BranchSystemAction
OperatorAT::infoChunking()	BranchAction
ProcedureStepUnit::executeAction()	BranchScannedParameter
ProcedureStepUnit::executeNonResponseAction()	BranchScannedParameter

After defining the PSFs, the crew failure modes with their mapping to the existing human events in ADS-IDAC, the next step is to obtain the data necessary to quantify the DBN. In order to achieve this, the estimated and calibrated parameters from the Phoenix method have been adopted. The data sources used in the Phoenix method include German NPP operating experience data, other

HRA methods (e.g. SPAR-H), and expert judgement. The advantage of the Bayesian network is that when new human performance data becomes available, be it qualitative or quantitative, it can be easily integrated into the model parameter estimation process using Bayesian inference. Common sources of information that can be used are experimental data (e.g. control room simulator data), operating experience (e.g., licensee event reports), HRA databases, like the US NRC sponsored database project called the Scenario Authoring, Characterization, and Debriefing Application (SACADA), in addition to expert judgement.

The conditional probability table for each crew failure mode node in the Bayesian network is used to capture the strength of influence between each crew failure mode and its parent PSF nodes. This implies that the probability of the crew failure mode given all its possible combinations of the PSFs needs to be defined. This is challenging problem as the number of conditional probabilities in the conditional probability table grows exponentially with the number of nodes and states.

To reduce the conditional probability table size, the noisy OR gates can be used to specify the DBN and to build the conditional probability table for the crew failure modes nodes. In relation to the DBN quantification model included in ADS-IDAC, the leaky noisy OR gate also give the advantage of representing the probability that a crew failure can occur even when there is no influence from any of the PSFs. In other words, the leak factor provides a way to include other PSFs that are not explicitly represented in the DBN model as individual PSF nodes.

It is important to note that data used in the Phoenix method has been calibrated to account for the different normalization schemes used to compile the data from the other HRA methods. Therefore, given that the calibration process has already been performed in the Phoenix method, it is no longer required. The final data transferred from the Phoenix method into the DBN of ADS-IDAC used to estimate the crew failure modes given a set of PSFs is given in Table 6-19.

Using the HAMMLAB empirical data and the HRA results from the international empirical study (Lois, 2009), the conditional probability table (Table 6-20) has been normalized using the following steps. The normalization procedure was the following: HFE 1A, failure to isolate the steam generator in the simple SGTR scenario, has been quantified using the original conditional probability table. The probability obtained has been scaled down two orders of magnitude such that the simulated HFE 1A falls inside the band of results obtained in the international empirical study.

Table 6-19 Conditional probability table and leak factors for each crew failure mode

Crew Failure Mode	**Performance Shaping Factors (PSFs)**										**Leak Factor**
	System criticality	Information load	Time constraint load	Cognitive task load	Passive alarm load	Expertise	Task complexity	Stress	Fatigue	Problem-solving style	
Perceive State	5.33E-3	2.69E-3	5.33E-3	5.26E-3	4.60E-3	2.45E-3	3.50E-3	4.29E-3	4.29E-3	2.89E-3	2.80E-5
Gather State Info	6.30E-3	3.17E-3	6.30E-3	6.21E-3	6.26E-3	2.90E-3	4.81E-3	2.71E-3	2.71E-3	3.42E-3	3.31E-5
Gather Info	1.67E-1	8.40E-2	1.67E-1	1.64E-1	1.66E-1	7.67E-2	1.27E-1	7.18E-2	7.18E-2	9.05E-2	8.75E-4
Gather New Info	5.28E-2	2.66E-2	5.28E-2	5.20E-2	2.29E-2	2.43E-2	4.03E-2	2.27E-2	2.27E-2	2.86E-2	2.77E-4
Diagnosis	2.71E-1	1.09E-1	2.71E-1	2.68E-1	2.06E-1	1.09E-1	2.36E-1	8.92E-2	8.92E-2	1.47E-1	1.42E-3
Strategy Selection	6.30E-2	2.52E-2	6.30E-2	6.21E-2	7.89E-3	3.68E-2	5.49E-2	2.71E-2	2.71E-2	3.42E-2	3.31E-4
Strategy Communication	3.78E-2	1.9E-2	3.78E-2	3.73E-2	3.75E-2	1.74E-2	2.89E-2	1.63E-2	1.63E-2	2.05E-2	1.98E-4

Crew Failure Mode		Maneuver Action	Procedure Interpretation	Procedure Step	Mental Procedure	Procedure Transfer	Goal Communi	Goal Selection
Performance Shaping Factors (PSFs)	System criticality	2.34E-2	1.54E-2	6.30E-2	8.91E-2	6.30E-2	3.78E-2	6.30E-2
	Information load	1.32E-2	6.17E-3	2.52E-2	3.56E-2	2.52E-2	1.9E-2	2.52E-2
	Time constraint load	2.34E-2	1.54E-2	6.30E-2	8.91E-2	6.30E-2	3.78E-2	6.30E-2
	Cognitive task load	5.17E-3	1.52E-2	6.21E-2	8.79E-2	6.21E-2	3.73E-2	6.21E-2
	Passive alarm load	4.77E-3	1.17E-2	4.78E-2	6.77E-2	4.78E-2	3.75E-2	7.89E-3
	Expertise	1.47E-2	9.62E-3	3.93E-2	5.55E-2	3.93E-2	1.74E-2	3.68E-2
	Task complexity	2.07E-2	1.97E-2	5.49E-2	7.76E-2	5.49E-2	2.89E-2	5.49E-2
	Stress	1.94E-2	6.65E-3	2.71E-2	3.84E-2	2.71E-2	1.63E-2	2.71E-2
	Fatigue	1.94E-2	6.65E-3	2.71E-2	3.84E-2	2.71E-2	1.63E-2	2.71E-2
	Problem-solving style	2.34E-2	8.38E-3	3.42E-2	4.84E-2	3.42E-2	2.05E-2	3.42E-2
Leak Factor		6.30E-4	8.10E-5	3.31E-4	4.68E-4	3.31E-4	1.98E-4	3.31E-4

Table 6-20 Normalized conditional probability table for each crew failure mode

Crew Failure Mode	Performance Shaping Factors (PSFs)										Leak Factor
	System criticality	Information load	Time constraint load	Cognitive task load	Passive alarm load	Expertise	Task complexity	Stress	Fatigue	Problem-solving style	
Perceive State	5.33E-5	2.69E-5	5.33E-5	5.26E-5	4.60E-5	2.45E-5	3.50E-5	4.29E-5	4.29E-5	2.89E-5	2.80E-7
Gather State Info	6.30E-5	3.17E-5	6.30E-5	6.21E-5	6.26E-5	2.90E-5	4.81E-5	2.71E-5	2.71E-5	3.42E-5	3.31E-7
Gather Info	1.67E-3	8.40E-4	1.67E-3	1.64E-3	1.66E-3	7.67E-4	1.27E-3	7.18E-4	7.18E-4	9.05E-4	8.75E-6
Gather New Info	5.28E-4	2.66E-4	5.28E-4	5.20E-4	2.29E-4	2.43E-4	4.03E-4	2.27E-4	2.27E-4	2.86E-4	2.77E-6
Diagnosis	2.71E-3	1.09E-3	2.71E-3	2.68E-3	2.06E-3	1.09E-3	2.36E-3	8.92E-4	8.92E-4	1.47E-3	1.42E-5
Strategy Selection	6.30E-4	2.52E-4	6.30E-4	6.21E-4	7.89E-5	3.68E-4	5.49E-4	2.71E-4	2.71E-4	3.42E-4	3.31E-6
Strategy Communication	3.78E-4	1.90E-4	3.78E-4	3.73E-4	3.75E-4	1.74E-4	2.89E-4	1.63E-4	1.63E-4	2.05E-4	1.98E-6

Crew Failure Mode		Maneuver Action	Procedure Interpretation	Procedure Step	Mental Procedure	Procedure Transfer	Goal Communi	Goal Selection
Performance Shaping Factors (PSFs)	System criticality	2.34E-4	1.54E-4	6.30E-4	8.91E-4	6.30E-4	3.78E-4	6.30E-4
	Information load	1.32E-4	6.17E-5	2.52E-4	3.56E-4	2.52E-4	1.90E-4	2.52E-4
	Time constraint load	2.34E-4	1.54E-4	6.30E-4	8.91E-4	6.30E-4	3.78E-4	6.30E-4
	Cognitive task load	5.17E-5	1.52E-4	6.21E-4	8.79E-4	6.21E-4	3.73E-4	6.21E-4
	Passive alarm load	4.77E-5	1.17E-4	4.78E-4	6.77E-4	4.78E-4	3.75E-4	7.89E-5
	Expertise	1.47E-4	9.62E-5	3.93E-4	5.55E-4	3.93E-4	1.74E-4	3.68E-4
	Task complexity	2.07E-4	1.97E-4	5.49E-4	7.76E-4	5.49E-4	2.89E-4	5.49E-4
	Stress	1.94E-4	6.65E-5	2.71E-4	3.84E-4	2.71E-4	1.63E-4	2.71E-4
	Fatigue	1.94E-4	6.65E-5	2.71E-4	3.84E-4	2.71E-4	1.63E-4	2.71E-4
	Problem-solving style	2.34E-4	8.38E-5	3.42E-4	4.84E-4	3.42E-4	2.05E-4	3.42E-4
Leak Factor		6.30E-6	8.10E-7	3.31E-6	4.68E-6	3.31E-6	1.98E-6	3.31E-6

6.3 Static strength of influence diagram

All the PSF nodes are connected to all the crew failure mode nodes, however depending on the conditional probabilities between them their causal relation varies considerably. In Figure 6-6 a graphical representation of the strengths of influence between the crew failure modes and PSFs is given. The arcs have different thickness depending on the strength of influence between the nodes that they connect. The strength of influence is calculated from the conditional probability table of the child node and essentially expresses some form of 'distance' between the probability distributions of the child node conditional on the state of the parent node. From the four measures of distance that could have been used for this graphical representation: Euclidean, Hellinger, J-Divergence, and CDF (Koiter, 2006), for Figure 6-6, the Euclidian was chosen since it proved to be most robust.

The strengths of influence have been normalized, with the thickest possible arc is given to that arc that has the highest strength of influence. The thicknesses of all other arcs are calculated proportionally to the thickest arc. Note that this is a static influence diagram that only makes use of the conditional probability tables present in the model and is, therefore, not context-dependent.

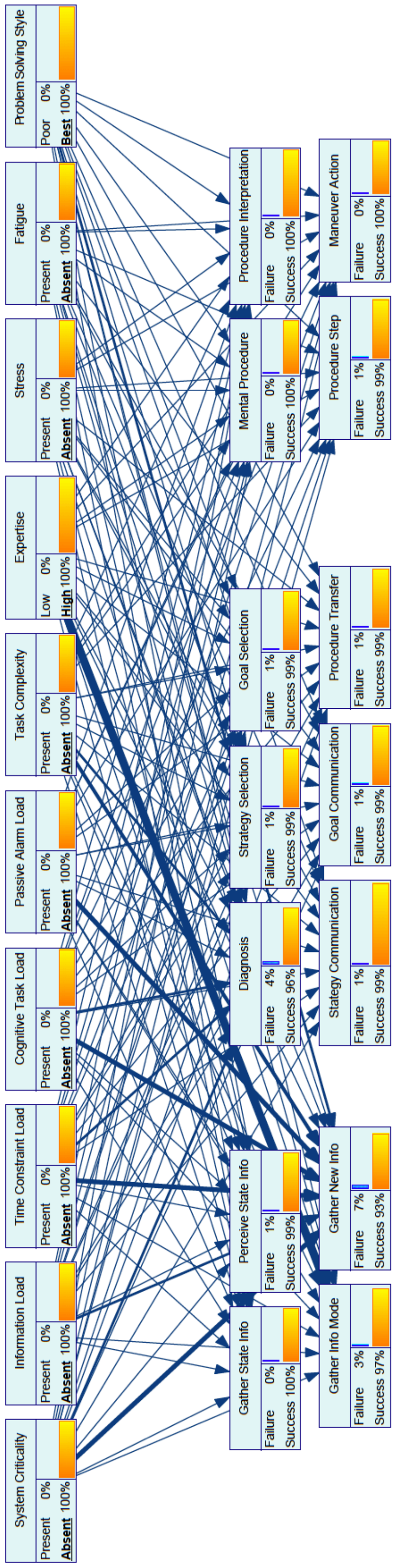


Figure 6-6 Static strength of influence diagram between the crew failure modes and the PSFs

7 Extended branching and uncertainty quantification of events in generation of DDETs

In this chapter, the new branching and quantification of events based on the newly developed dynamically linked FTs (for frontline and support systems) and DBN (for human events) are introduced. Moreover, the propagation of aleatory uncertainties through the resultant DDET is described based on the quantification models and sampling techniques already introduced in chapter 4.

7.1 Extended branching in ADS-IDAC

Previous research efforts in developing ADS-IDAC have shown that a small set of generic branching rules are sufficient to capture complex variations in system and crew-to-crew performance. The ADS-IDAC simulation engine generates a DDET by activating success, failure or partial failure branching points when certain conditions are met. All the events that occur between the initiating event and the branching points, or between the branching points are called intermediate events. Overall, all the branching points and the intermediate events associated with an initiating event make up the DDET during accident scenarios. The set of generic branching rules cannot create the DDET without modeling in parallel the dynamic system and human operator behaviors. Therefore, the construction of the DDET is driven by a rich contextual environment simulated by ADS-IDAC and guided by the branching rules, which allow the modeling of variability in system and human operator.

The necessity for branching rules in a dynamic PRA simulation platform like ADS-IDAC is the sequence explosion phenomenon. If the simulation engine would allow branching at every time step the number of sequences needed to be explored would grow exponentially and the simulation time would become unrealistically long with the current computational models and resources. For the same computational reasons, sequence termination conditions have been implemented to stop the engine from exploring sequences after a time period of interest, when certain physical limits have been exceeded, or when the operators enter certain procedures. For example, if the interest of the simulation is the exploration of crew variability in diagnosing a SGTR, two sequence termination conditions could be set to stop the simulation. One of them could be placed when the operators transfer to procedure E-2 “Isolation of steam generator with secondary break” or E-3 “Tube rupture in one or several steam generators.” Another could be set when the simulation time exceeds a certain time period set based on previous crew performance.

At the same time, sequence termination conditions can be set to calculate an overall failure probability for an event of interest. For example, a sequence termination condition for the fuel element cladding temperature exceeding the acceptance criteria for emergency core cooling systems for light water nuclear power reactors (10 CFR Part 50.46) of 2200° F. The summation of the end state probability for all sequences that were terminated by this condition would essentially estimate an overall measure of core damage probability.

Overall, the branching rules and the sequence termination conditions help define the scope of the intended ADS-IDAC analysis. Therefore, if the set of branching rules and sequence termination conditions do not cover all the models included in ADS-IDAC their variability is not included in the generated DDET and, ultimately, the solution space is not complete.

The linking of dynamic FTs has been described in chapter 5 and illustrated again in Figure 7-1. Therefore, branching rules that cover the failure of either frontline or support system components have also been implemented. Nonetheless, by implementing branching rules alone does not mean the sequence end state probability can be quantified. Each branching point requires either a success or failure probability. The ADS-IDAC hardware reliability module was extended to cover modeling of both frontline and support system failures during operation by considering the failure rate, the number of failures desired and the time interval between them. DDET branching is modeled such that failures during operation generate two branches: success and failure branch. For a specific equipment, if more than one failure during operation is modeled, only the subsequent success branches will further allow more failures as on the failure branches this equipment had already failed. This feature further extends ADS-IDAC's capability to dynamically predict the timing importance of component failures during operation for the overall safety of the design in question is. For example, small-break LOCA scenarios in a PWR could be set up to simulate the timing of the pressurizer power operated relief valves (PORVs) stuck-open failure events and their impact on the available time for recovery actions.

The crew's recovery of frontline and support system's component failures can also be modeled. When the crew attempt to recover a component, two additional branches are generated: a recovery branch, in which the component is successfully recovered, and a permanent failure branch, in which the component remains failed.

The branching points that were quantified with the dynamically linked FTs, out of which a success and a failure branch are created, are given below:

- Components of frontline systems failures at fixed time.
- Components of frontline systems failures on demand.
- Components of frontline systems failures during operation.
- Components of support systems failures at fixed time.
- Components of support systems failures on demand.
- Components of support systems failures during operation.

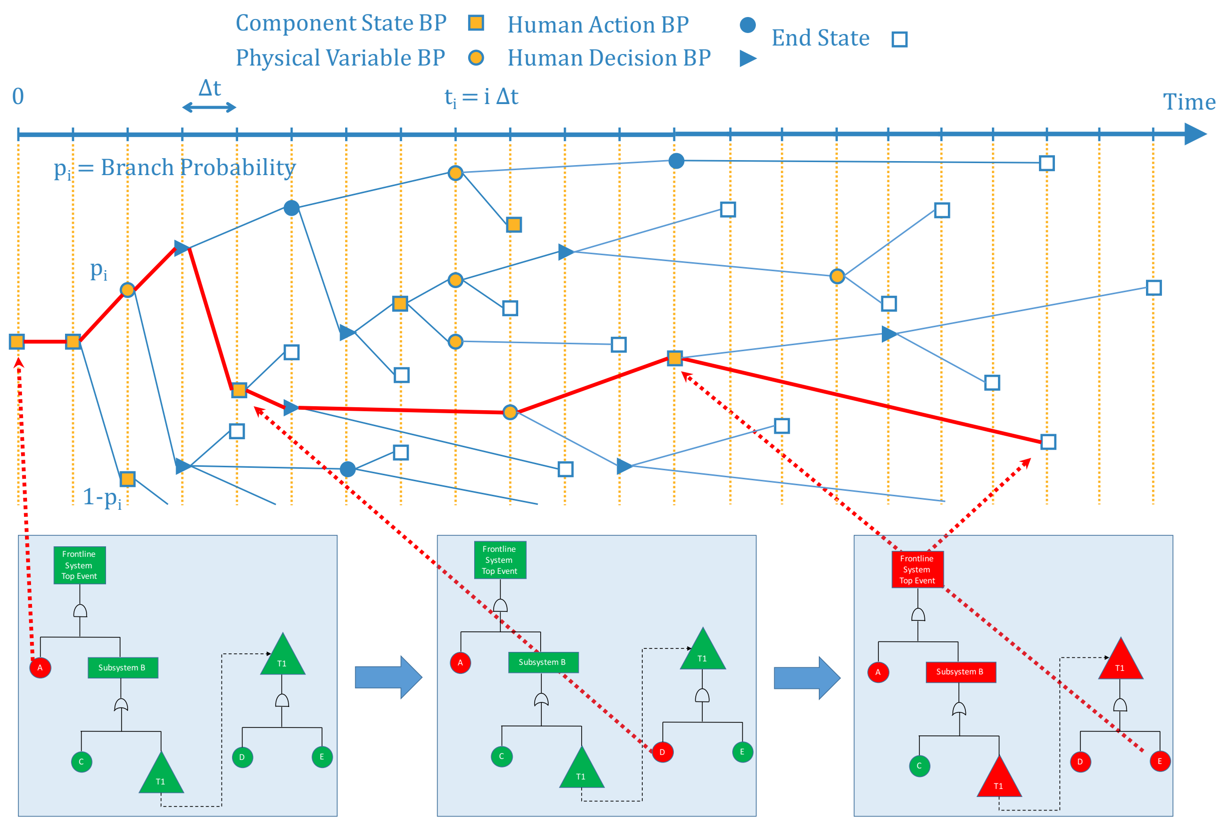


Figure 7-1 DDET with quantified frontline and support system branching points based on dynamically linked FTs

One of the key goals for the ADS-IDAC project was to develop a simulation-based human reliability analysis tool capable of identifying contextual factors that could lead to human error events. A primary focus of previous studies for DDET branching was capturing a wide range of potential human performance variability, but not so much on the quantification of those branching points. In this research, each branching rule probability has been mapped to the hardware and human events calculated probabilities at each time step. Given that the quantification of the human events in modeled with a DBN, the probability for an operator to perform a human event is explicitly defined in terms of the PSFs that reflect the current state of the context (Figure 7-2). One consequence of this framework can be seen with the following simple example. During a diagnosis, the crew may need to check the status of a component multiple times. Depending on the context, and implicitly on the value of the PSFs, the probability of the crew to correctly perceiving the status of that same component may be different as it is graphically represented in Figure 7-2.

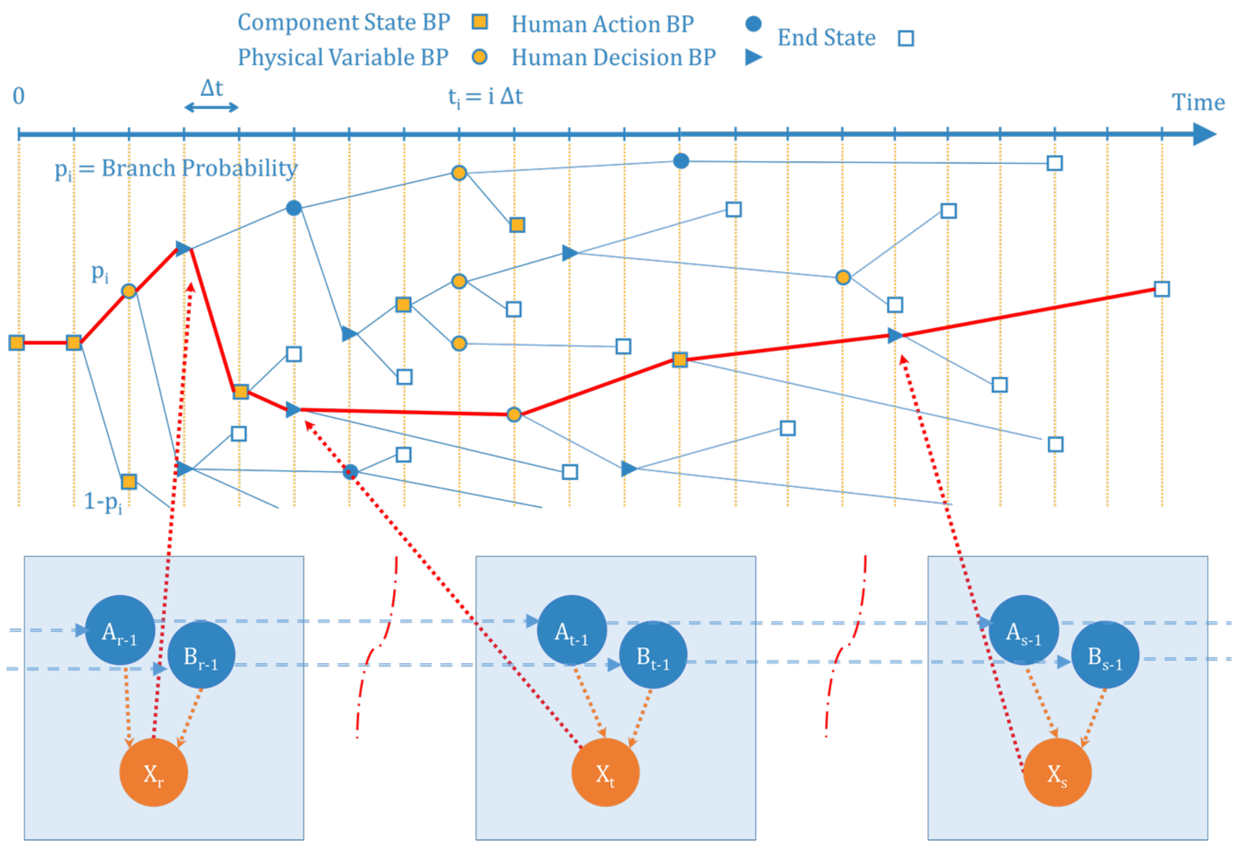


Figure 7-2 DDET with quantified human events branching points based on a DBN

The branching points that were linked to be quantified with the DBN are described below:

- When a strategy is changed, based on the "Strategy Selection" crew failure mode two branches are generated: one in which the crew continue the current strategy, and another in which they switch to the new strategy.

- When a procedure step indicates a transfer to another procedure, two branches are generated: one in which the crew switches to the new procedure, and another in which they continue the current procedure.

- When an accident diagnosis threshold is exceeded based on the knowledge-based reasoning, two branches are generated: one in which the crew take recovery actions based on their reasoning, and another in which they transfer to the appropriate procedure.
- When a mental belief activation threshold is exceeded based on the heuristic reasoning two branches are generated: one in which the crew transfer to the mental belief, and another in which the crew bypass the mental belief and continue their activity.

These rules together with the newly developed quantification models help to keep the simulation space expansion under control, yet it also allows sufficient degrees of freedom for the system and crew to evolve into unexpected behaviors. For example, given the procedure step skipping probability is quantified at each time step containing written procedure steps or mental procedures either deterministically or stochastically, the skipping of procedure steps could be simulated and their impact analyzed in a consistent and transparent way.

A full quantification of the DDET was achieved that includes not only the human failure and success events as branching points, but also all the successful intermediate events. This is a very critical aspect of the full DDET quantification as now all the successful human events have a probability covering the full unit interval instead of a fixed probability of one.

7.2 Uncertainty quantification in ADS-IDAC

In this section, the techniques used to propagate the aleatory uncertainties through the DDET are described. When the probabilities of the initiating, branching, or intermediate events in a DDET are subject to uncertainty, the probabilities can be considered to be random variables described by some probability distribution. The form of their probability distribution depends on the type of events (e.g., component failure, human activity, etc.) Therefore, the probability of the end state events in such a DDET will also be a random variable, and the form of its probability distribution will depend both on the DDET structure and the probability distributions of the events. In this research, various sampling techniques (i.e. MCS, LHS, QMCS) are first implemented in an updated version of HCL and then ported into ADS-IDAC. The sampling techniques have already been described in chapter 4. They are used for the propagation of aleatory uncertainties in the DDET generated by ADS-IDAC. These Monte Carlo methods are used to obtain a probability distribution of the end state events in a DDET using available information on the tree structure and the assumed probability distributions of its top events. The same methods can be applied to propagate the uncertainties through the FTs used to represent frontline and support systems. For these accident sequences, the propagation of uncertainties is performed on the combined structure of DDETs and FTs.

Previously, ADS-IDAC employed the best estimate propagation where the events are represented only by their best estimate. In this research, the events are represented by a certain probability distribution function and through Monte Carlo methods samples of these events probabilities are propagated through the DDET.

The DDET contains multiple types of events: initiating event, branching events, intermediate events and end state events. In a typical DDET, the number of intermediate events is multiple orders of magnitude higher than the number of branching events. The uncertainty in the probability of the branching events is more critical than of the intermediate events as they include low probability failure events that lead to accident conditions. To the number of random variables considered, only point estimates of intermediate event outcomes are used.

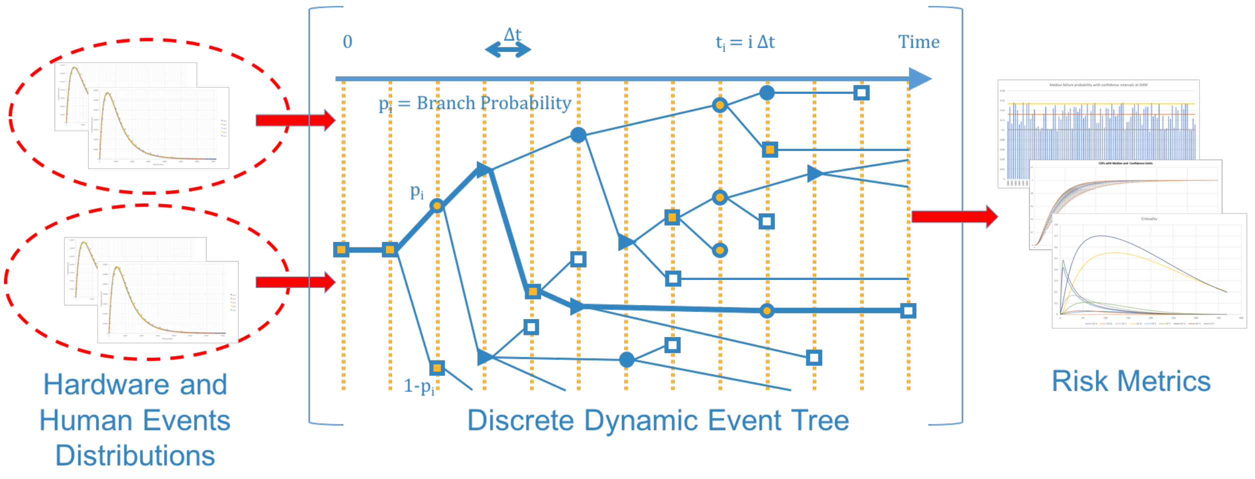

0
Δt
$t_i = i\,Δt$
Time
p_i = Branch Probability
p_i
$1-p_i$
Hardware and Human Events Distributions
Discrete Dynamic Event Tree
Risk Metrics

Figure 7-3 Uncertainty propagation of aleatory uncertainties in ADS-IDAC

The ADS-IDAC simulation with uncertainty quantification is performed in two stages. First, the DDET is initially generated and quantified obtaining a best estimate end state probability for each sequence. Second, the end state uncertainties are obtained by performing a Monte Carlo simulation using information about the scenarios, the type of initiating event or branching events with their associated best estimates or probability distributions used to model their variability. (Figure 7-3) The initial stage of the ADS-IDAC simulation uses mean values to obtain the end state probabilities, while the second stage involves the stochastic layer to propagate the uncertainties all the way to the end states. It is recognized that the decision to propagate the uncertainties in this phased approach is a first approximation due to the computational demands of the thermal-hydraulic code.

Finally, in the ADS-IDAC hardware reliability module, the user can now select from among the common, non-parametric or custom probability distributions with global variables. The HFEs probabilities are sampled from the log-normal distributions with the mean calculated during the point estimate phase using the DBN of the PSFs as described in the previous section and the variance selected either from the literature or using expert judgement.

8 Test case: Analysis and quantification of a PWR steam generator tube rupture event

In this chapter, a test case capturing the NPP and crew behavior given a concurrent steam generator tube rupture (SGTR) and a main steam line break (MSLB) designed for the International HRA Empirical Study (Lois, 2009) was simulated to showcase the new models implemented into ADS-IDAC.

8.1 Description of the steam generator tube rupture complex scenario of the International HRA Empirical Study

The test case set up is based on the SGTR complex scenario selected from an International HRA Empirical Study (Lois, 2009). The scope of the empirical study was to perform experiments at the Halden Reactor Project's HAMMLAB (HAIden huMan-Machine LABoratory) research simulator where real crews were asked to respond to a series of carefully designed accident conditions to build an empirically based understanding of the performance, strengths and weaknesses of the most used conventional HRA methods.

The HAMMLAB research simulator is a three-loop Westinghouse PWR. Also, The HAMMLAB's EOPs were loosely based on the emergency response guidelines (ERGs) developed by the Westinghouse Owners Group. The EOPs used in the complex SGTR scenario are: E-0 – "Reactor Trip or Safety Injection" and E-3 "Tube rupture in one or several steam generators". E-0 is the safety systems verification and diagnosis procedure used by the crew when the reactor has tripped, when safety injection has started, or when there is a need for either of them. E-3 is the procedure to which the crew are typically expected to transfer from E-0 when a diagnosis of SGTR is declared and contains the recovery instructions. The crew have other procedures from which they can transfer to E-3, or they can use their knowledge only to proceed to E-3 if the other procedures give them conflicting instructions. To limit the scope of this complex SGTR scenario, only the E-0 and

E-3 are considered. Two steps in procedure E-0 are relevant for the test case set up to be simulated: step 19 – which is the first step where the crew can transfer to procedure E-3 in response to elevated secondary radiation indications of a SGTR, and step 21 – which is the second step where the crew can transfer to procedure E-3 based on any of the steam generator's level rising uncontrollably.

The complex SGTR scenario starts with a concomitant SGTR and a main steam line break (MSLB) in normal operation at 100% that immediately activates automatic SCRAM and the expectation that the crew with open the E-0 procedure for verification of safety systems and diagnosing the NPP condition. Two more complications were introduced to create a more complex scenario: the main steam isolation valves (MSIVs) close automatically in response to the MSLB, and the failure of any remaining secondary radiation indications. The scenario was designed so that as the MSLB drives the NPP response early in the scenario, where the initial symptoms of the NPP resemble a severe MSLB with the quick closure of the MSIVs, and all the secondary radiation indications fail. These conditions are expected to mask the occurrence of the SGTR and make its diagnosis a lot more challenging than the typical SGTR alone where the crews are expected to transfer to E-3 at step 19. This challenges the crew's procedure-following strategy, and a correct diagnosis of the NPP conditions would heavily rely on their knowledge-based reasoning or otherwise delay the transfer until they reach step 21.

In Figure 8-1, the ET for this SGTR scenario is illustrated to show the HFEs labeling and the top events. The HFE selected to be quantified in ADS-IDAC is HFE 1B: failure of the crew to isolate the faulted steam generator. The top events of interest leading to HFE 1B are: successful reactor trip, feedwater available, and high-pressure injection started. To successfully isolate the faulted steam generator, the crew need to successfully perform the following activities:

- Transfer from E-0 to E-3

- Verify steam dump to atmosphere valve set point is above 70.5 bar.
- Verify blow down isolated.
- Verify main feedwater isolated.
- Verify atmosphere valve's steam dump closed.
- Close steam valve to turbine-drive auxiliary feedwater pump.
- Verify lock steam valve to turbine-drive auxiliary feedwater pump closed.
- Verify steam traps closed.
- Close MSIV and its bypass valve.
- Stop auxiliary feedwater to the ruptured steam generator when the narrow range steam generator indicator is greater than 10%.

Compared to the conventional HRA methods, which quantify the HFE in its entirety, ADS-IDAC is able quantify each individual activity that can lead to that particular HFE. Therefore, it is not only able to transparently predict the system and crew behavior, but quantify the probability of succeeding or failing at each time step for each activity the crew is undertaking based on the actual context.

Although only the performance of one crew was analyzed in this chapter, for completeness, in Table 8-1, the 14 crews' procedure progression and their basis for transferring to E-3 in the SGTR complex scenario are given. 5 out of 14 crews found the basis for transferring to E-3 by following the procedures, while the other 9 transferred to E-3 based on their knowledge-based reasoning.

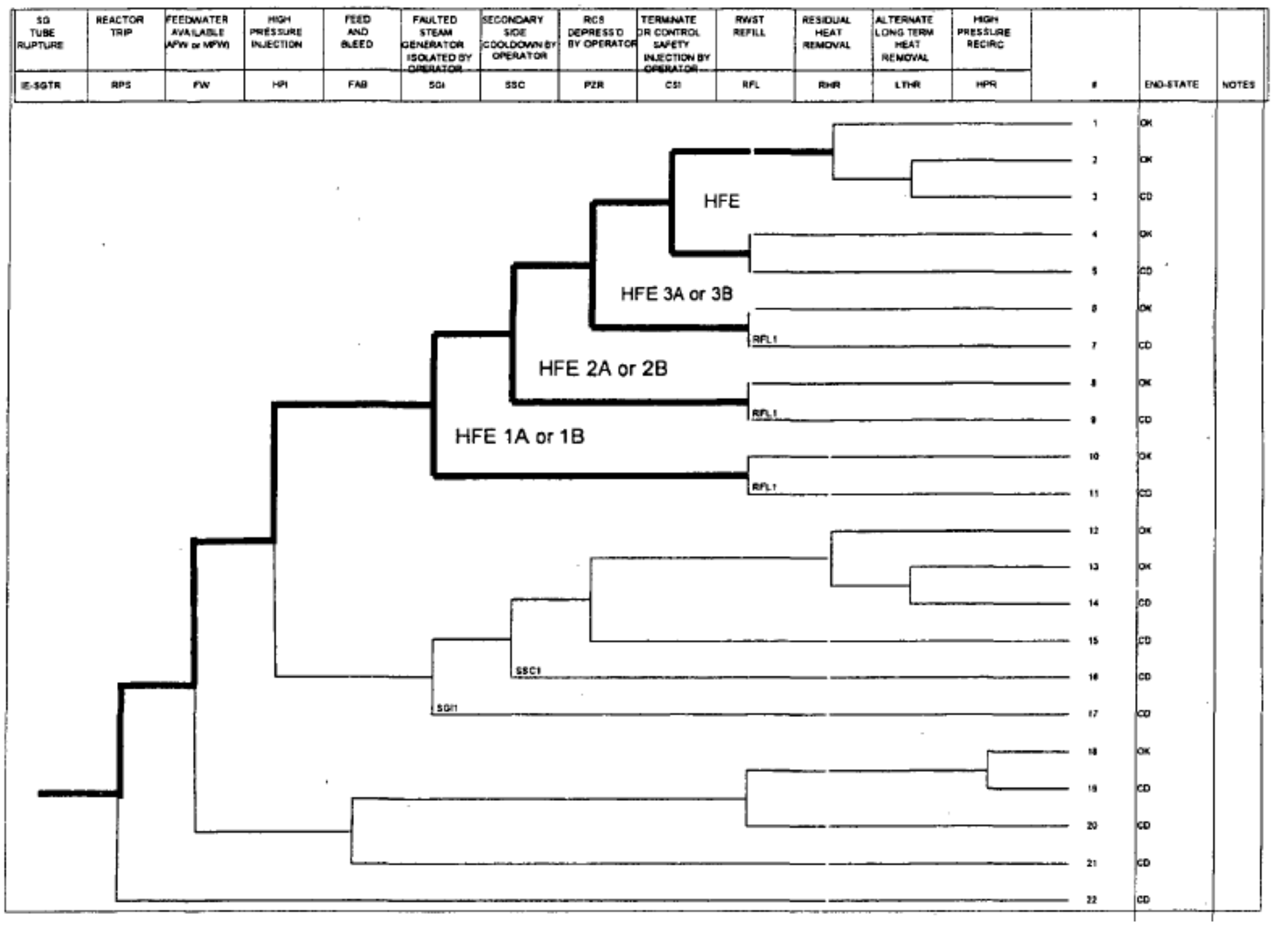


Figure 8-1 Typical PRA ET for a steam generator tube rupture scenario (Lois, 2009)

The sequence of interest for the selected test case is replicated only by crew M, which diagnose the SGTR based on the steam generator level at step 21 by following the procedures.

Table 8-1 Procedure progression and basis for transfer to E-3 in the SGTR complex scenario

Crew	Point of transfer to E-3	Basis for transfer to E-3
A	E-0 step 21 – ES-1.1 foldout page	SG level
B	E-0 step 24	SG level
C	E-0 step 21	Knowledge-based (level)
D	E-0 step 24-25	Knowledge-based (level)
E	E-0 step 21 – ES-1.1 – E-0 step 19	SG1 gamma levels 1 and 2 (slow crew)
F	E-0 step 21 – ES-1.1 – E-0 step 19	Knowledge-based (level)
G	E-0 step 21	Knowledge-based (level)
H	E-0 step 21 – ES-1.1 – FR-H5 – E-0 step 19	Knowledge-based (level)
I	E-0 step 21 – ES-1.1 –E-0 step 19	Knowledge-based (level)
J	E-0 (second loop) step 14 – E-2 step 7	Knowledge-based (level)

K	E-0 step 19	Gamma radiation
L	E-0 step 21	Knowledge-based (level) + ES-1.1 foldout
M	E-0 step 21 – ES-1.1 foldout page	SG level
N	E-0 step 21	Knowledge-based (level)

For uncertainty quantification in ADS-IDAC, a probability distribution and its variance is needed for the HFEs given that the means are obtained in the best estimate stage of the dynamic simulation. In this respect, the results of the empirical study provide a good basis. Therefore, a lognormal distribution has been selected for all the HFEs. From Figure 8-2 of HFE 1B estimation using various HRA methods, the variance in probabilities of all the HFEs is around 1.5E-3.

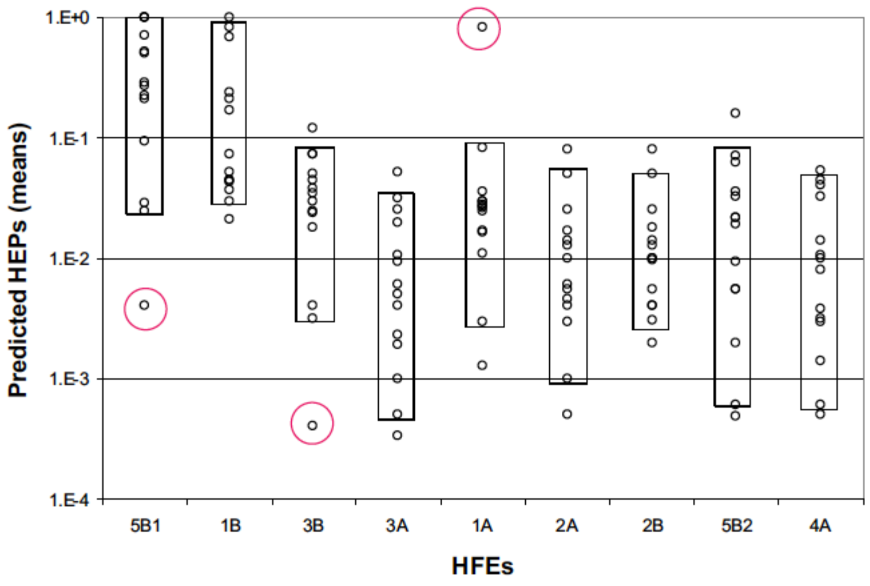


Figure 8-2 Range of predicted mean HEPs of the HRA methods (Lois, 2009)

8.2 ADS-IDAC simulation model

The SGTR scenario simulation model[3] used to showcase the new features implemented in ADS-IDAC is based on Yuandan Li's model replicating the SGTR complex case scenario of the

[3] The ADS-IDAC .ads file can be obtained on request or it can be found in the "testcases" folder that comes with the .jar application.

International HRA Empirical Study (Li, 2013). It includes a generic three-loop PWR thermal-hydraulic model that mimics a Westinghouse PWR using RELAP5/MOD 3.3.

Procedures E-0 and E-3 mentioned in the previous section are modeled to drive the procedure-following strategy of the crew necessary in the SGTR scenario. The important difference between the HAMMLAB's procedures and ADS-IDAC's procedures for this scenario is that E-0 Step 19 in HAMMLAB corresponds with E-0 Step 16 in ADS-IDAC, and E-0 Step 21 in HAMMLAB corresponds with E-0 Step 19 in ADS-IDAC. The difference is because of three missing steps in ADS-IDAC's E-0. Also, given that the operator's procedure-following path is greatly challenged, the crew is expected to heavily rely on the knowledge-based reasoning for a correct diagnosis of the accident. For this reason, the simulation model includes a knowledge base created specifically for this type of accident. It contains the knowledge links, and the concepts for parameter indicators and alarms necessary for a correct diagnosis and recovery from a SGTR scenario.

This test case is not intended to further evaluate or expand the ADS-IDAC operator's response to the abnormal conditions. The test case was selected to showcase the quantification of all the events in the DDET and uncertainty quantification with the new version of ADS-IDAC.

8.3 ADS-IDAC simulation results

The results obtained from the ADS-IDAC simulation are described below. Note that although ADS-IDAC generates multiple sequences and covers most of the crew variability observed in the international empirical study, the results given here cover only one of the sequences, namely the one described in chapter 8.1 of crew M.

In Figure 8-3, the exponential decrease in time of the core power is given. This was expected as the reactor trips automatically because of the MSLB.

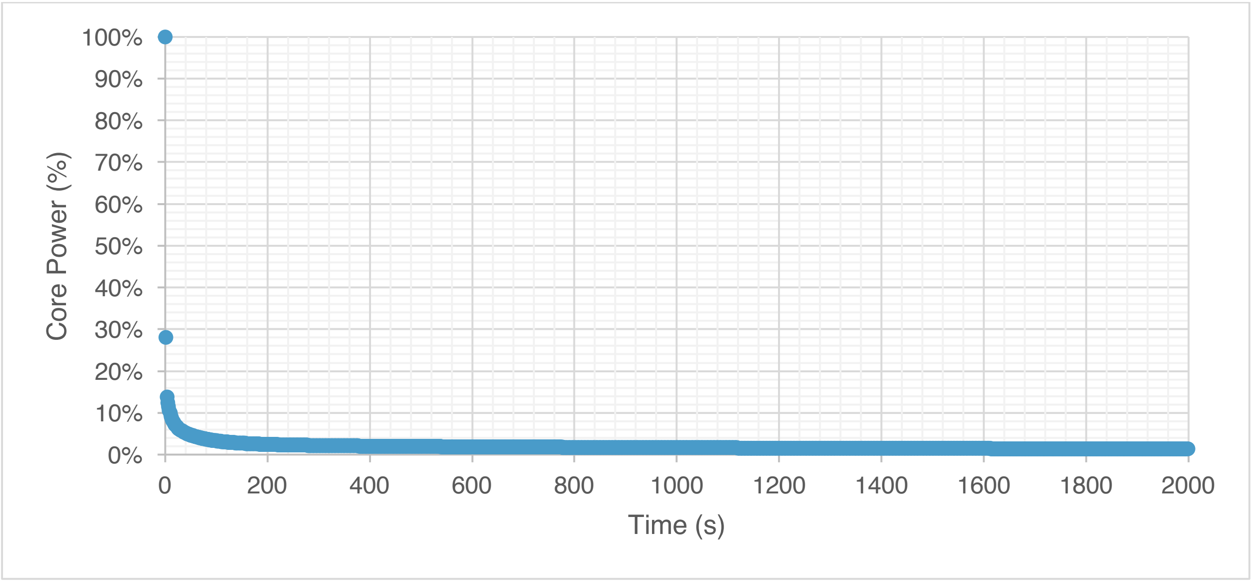


Figure 8-3 Core power evolution

In Figure 8-4 the pressurizer pressure evolution is illustrated. It can be seen the abrupt fall in the beginning of the scenario as the main steam line breaks. After the MSIVs close, the pressurizer pressure starts to recover. The pressurizer pressure is also increasing because of the safety injection actuation.

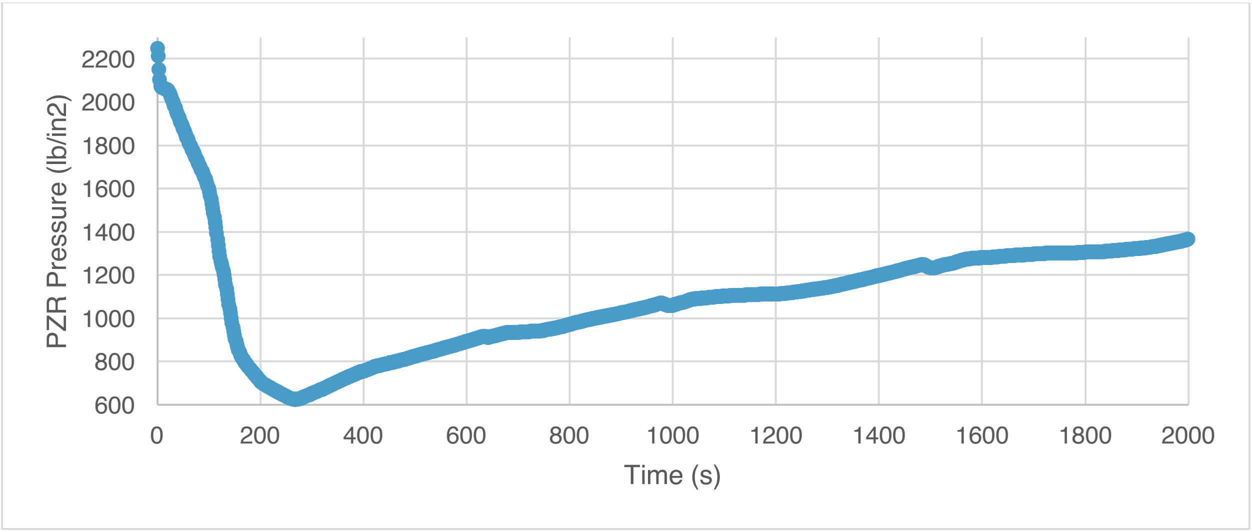


Figure 8-4 Pressurizer pressure evolution

In Figures 8-5 and 8-6, the steam generators narrow range level and, respectively, the wide range level evolution are shown. The narrow range level dropped sharply at the beginning of the simulation as the reactor tripped. After 800 seconds into the scenario, steam generator A narrow range level is much greater than the levels of steam generator B and C. This is a critical piece of information that can help the crew diagnose the SGTR. The level unbalance between steam generator A and steam generators B and C could only be explained by having a source of water into steam generator A. After checking all the water sources, the crew could diagnose the SGTR of steam generator A. Nonetheless, in the sequence analyzed here, around 300 seconds into the scenario, the crew noticed that the wide range level of steam generator A was increasing faster than the other two although the auxiliary feedwater was closed. Based on this, the crew decided to transfer to procedure E-3 to isolate the faulted steam generator.

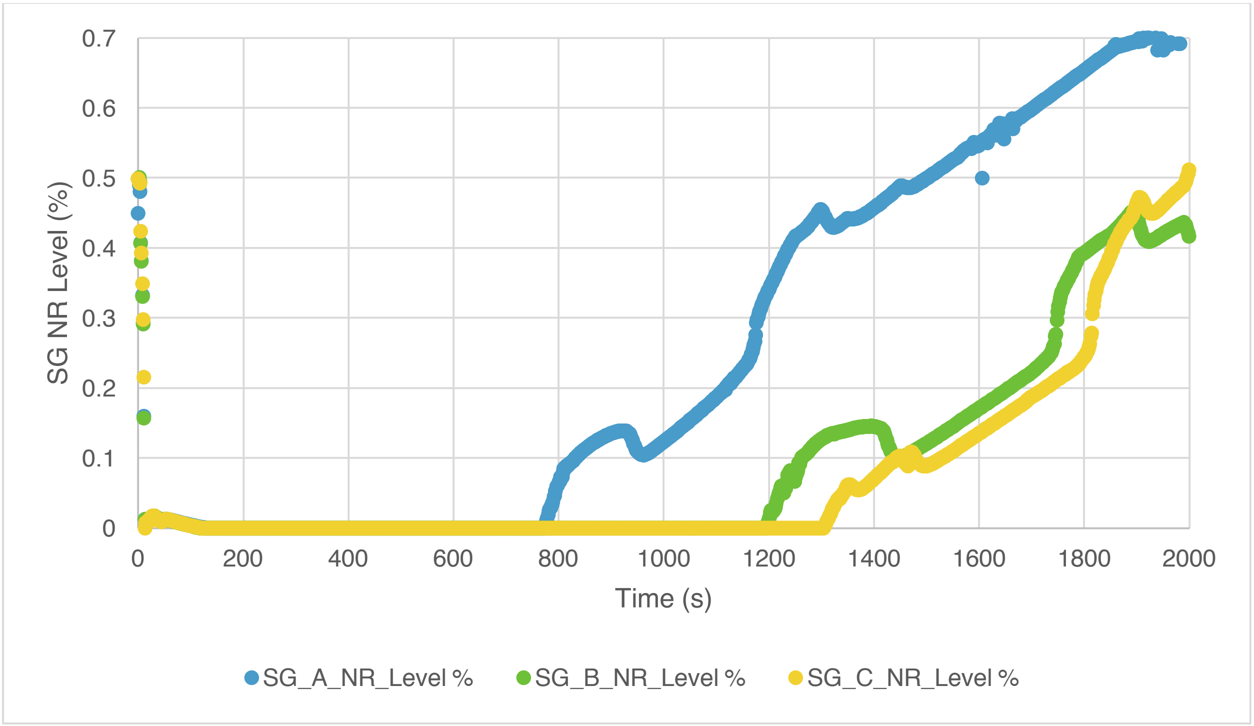


Figure 8-5 Steam generators narrow range level evolution

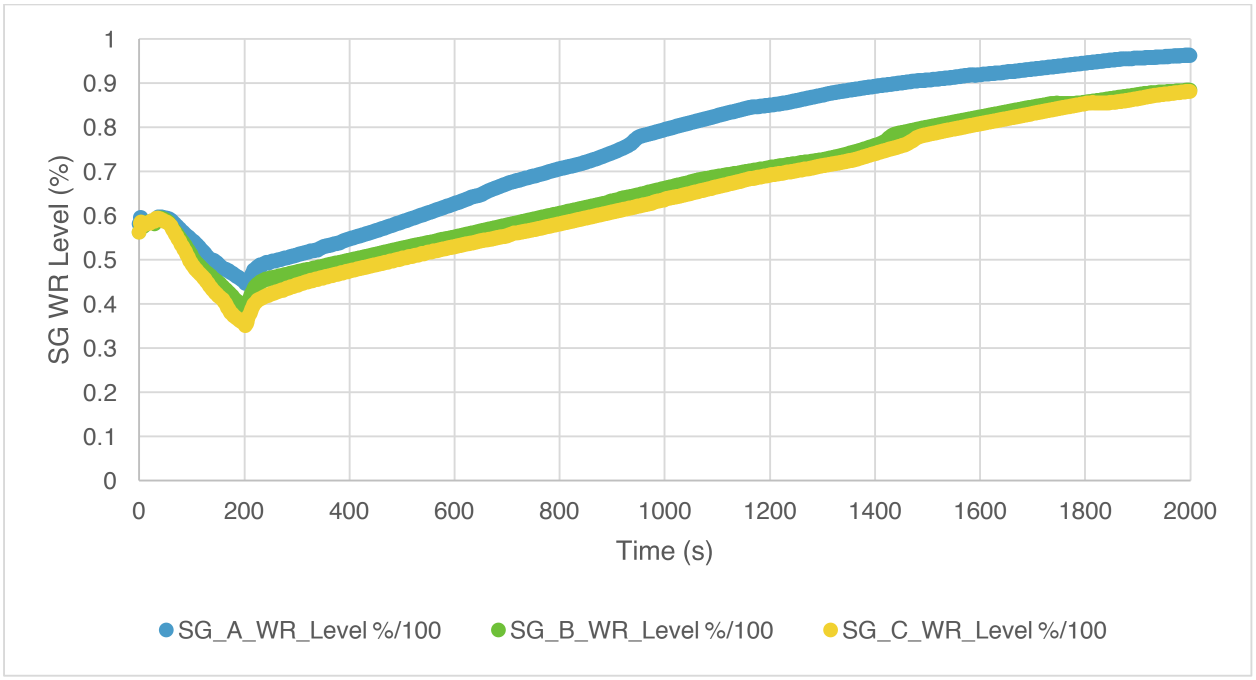


Figure 8-6 Steam generators wide range level evolution

In Figure 8-7, the steam generators pressure is plotted. After the reactor is tripped, the steam generator pressure increases as expected. But because of the MSLB and SGTR, the pressure drops rapidly. Both scenarios have similar symptoms, thus the crew would not be able to declare a diagnosis so early in the sequence. However, after steam generator A is isolated, its pressure increases as can be seen after 1300 seconds because of the RCS leakage through the ruptured tube.

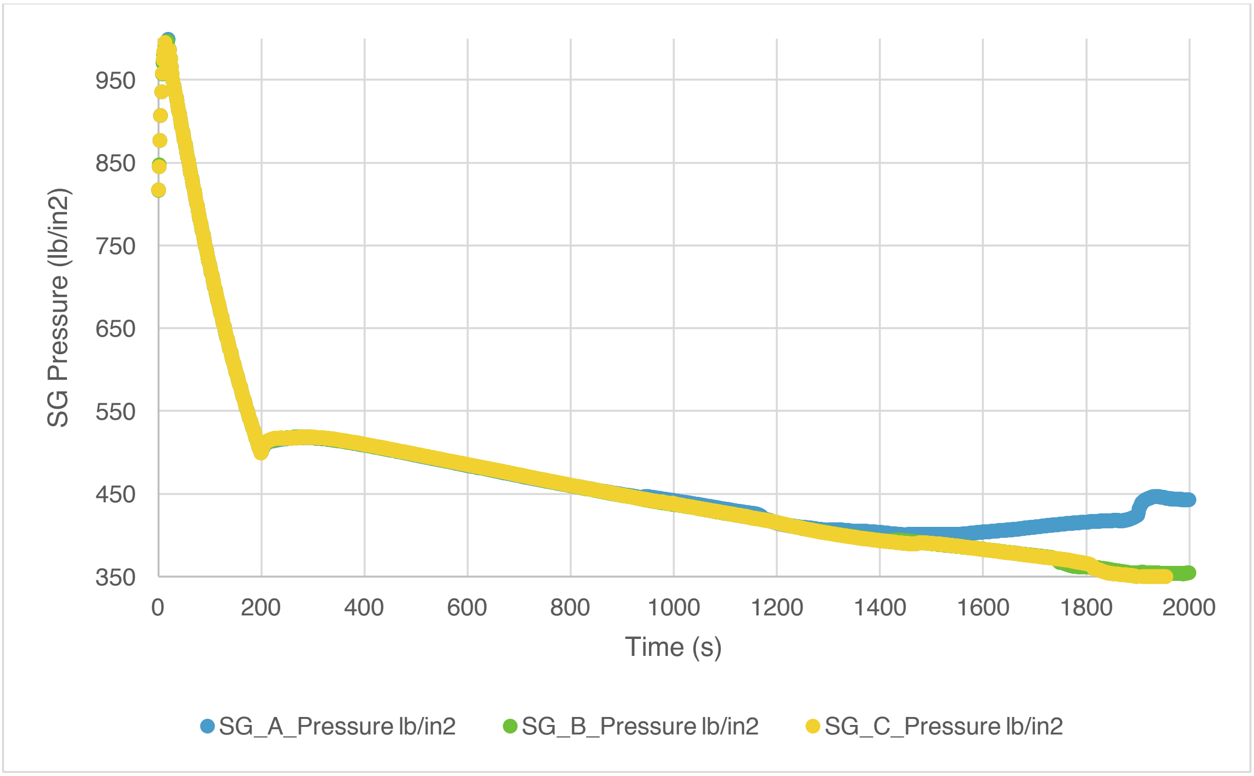


Figure 8-7 Steam generator pressure

In Figure 8-8, the decision-maker's dynamic PSFs evolutions are illustrated together. By looking at the passive alarm load, cognitive task load, diagnosis complexity and stress around 300 seconds, it can be seen that the decision-maker is busy diagnosing the abnormal conditions. Right after diagnosis of the SGTR, the stress and diagnosis complexity decrease and level off, while the cognitive task load remains high because of activities related to the isolation of the faulty steam generator.

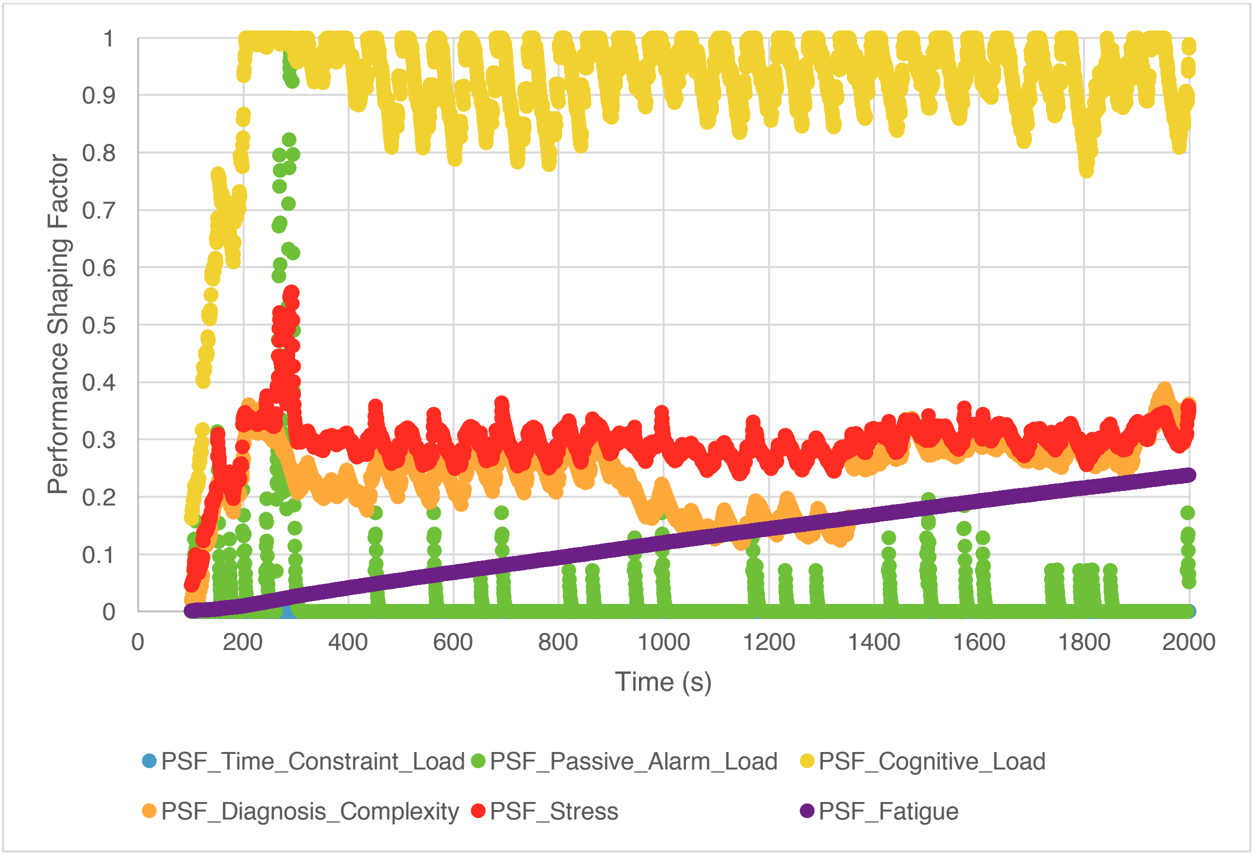


Figure 8-8 Decision-maker's PSFs evolution

In Figure 8-9, the decision-maker's and action-taker's passive information load are compared. While the decision-maker and the action-taker are very busy at the beginning of the simulation perceiving all the alarms and parameters, the action-taker's information load gradually decreases while the decision-maker is still quite active in diagnosing the fault.

In Figure 8-10, it can be seen that the action-taker's time constraint load is very high at the beginning of the simulation as the crew need to perform many actions in a short time period. The jump at around 1500 seconds is due to the throttling of the motor-driven auxiliary feedwater to steam generator A as its pressure was starting to increase compared to the other two steam generators.

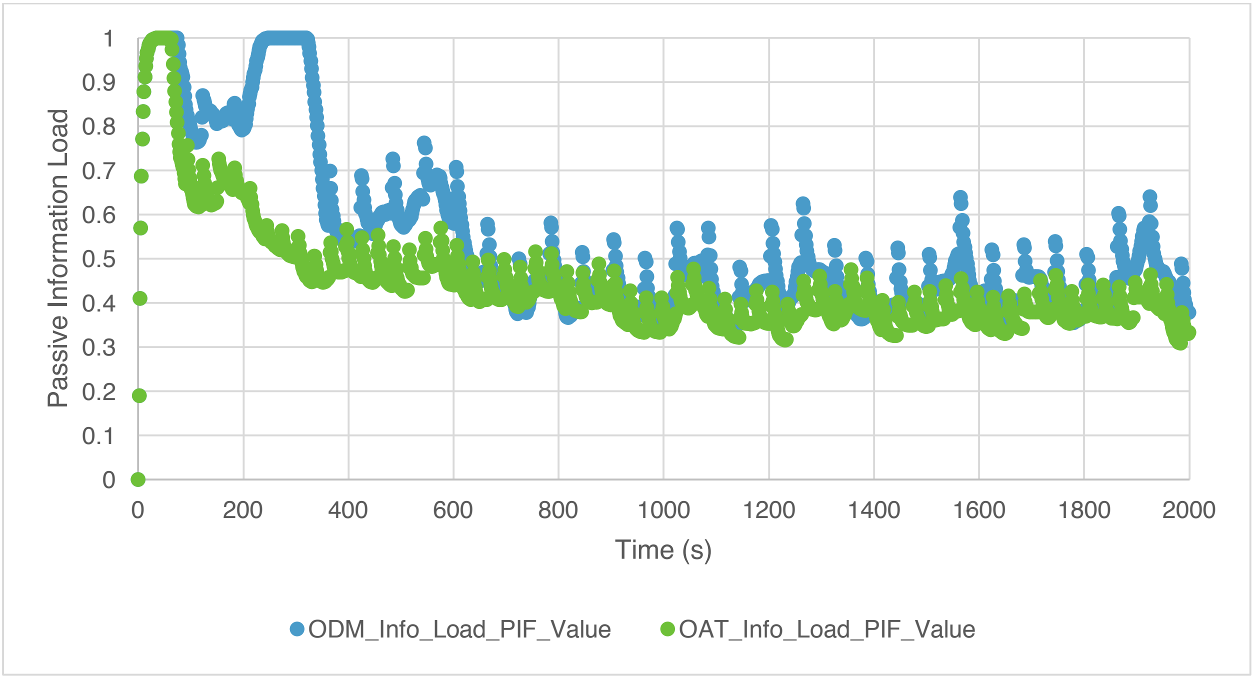


Figure 8-9 Comparison between the passive information load PSF evolutions of the decision-maker and action-taker.

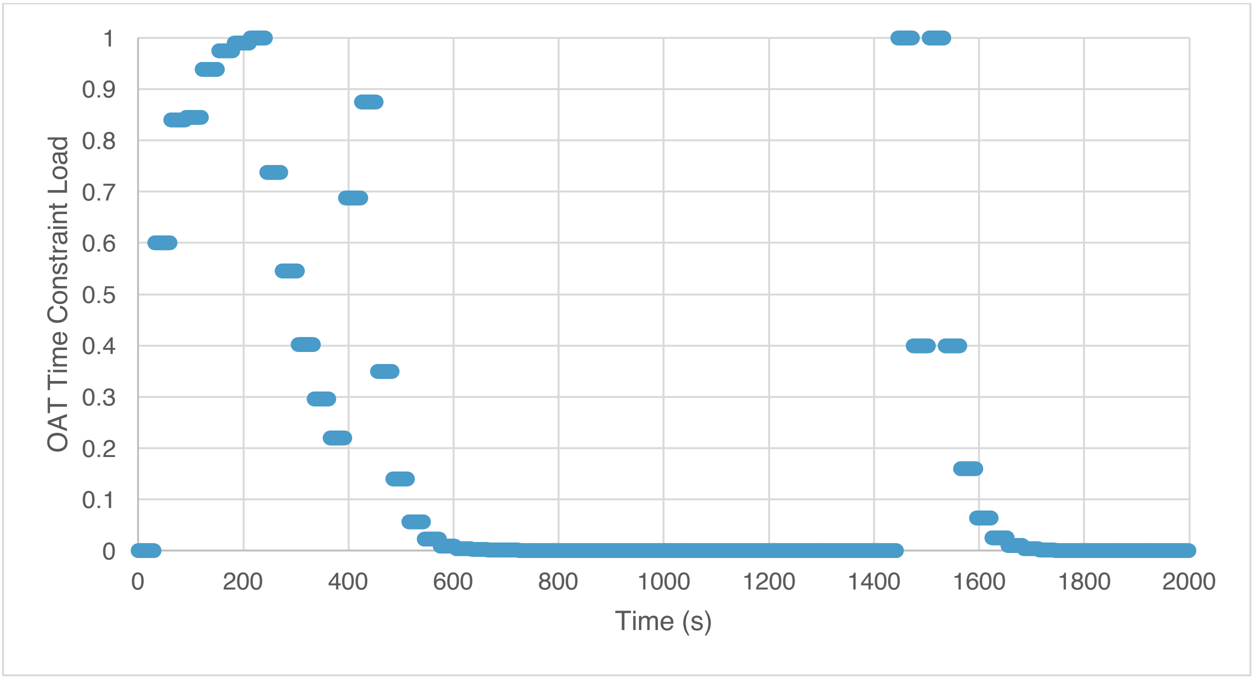


Figure 8-10 Action-taker's time constraint load PSF evolution

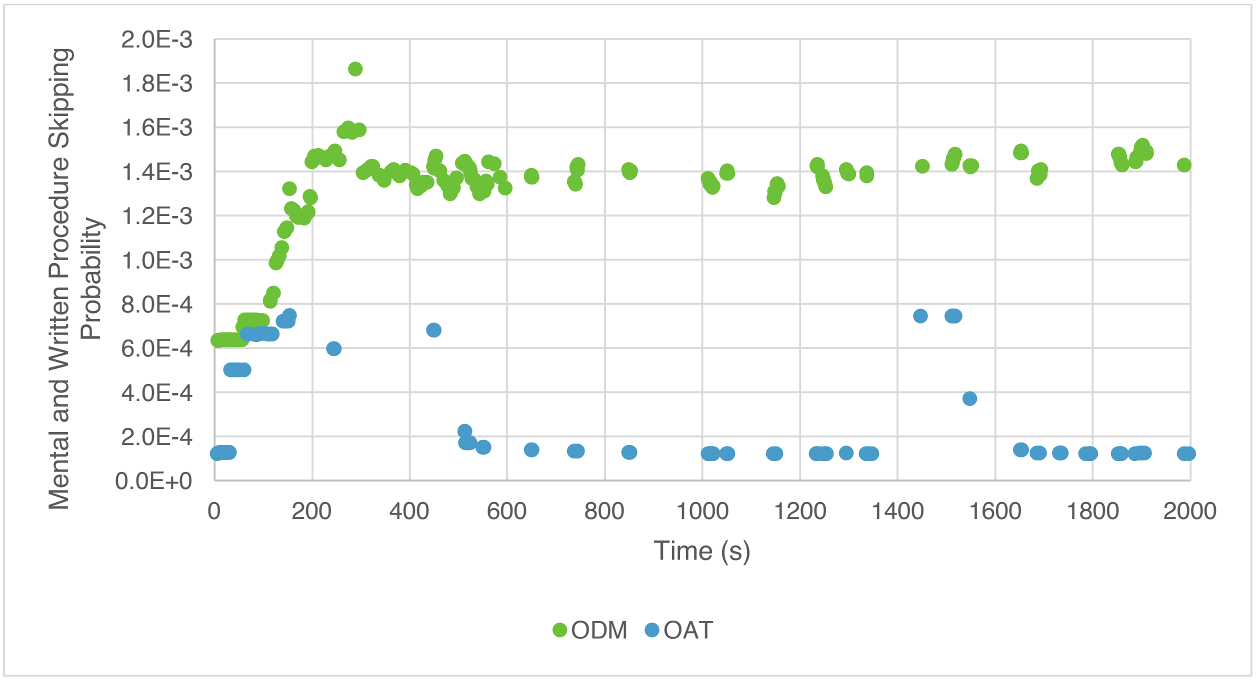


Figure 8-11 Comparison between the decision-maker's and action-taker's mental and written procedures step skipping probability

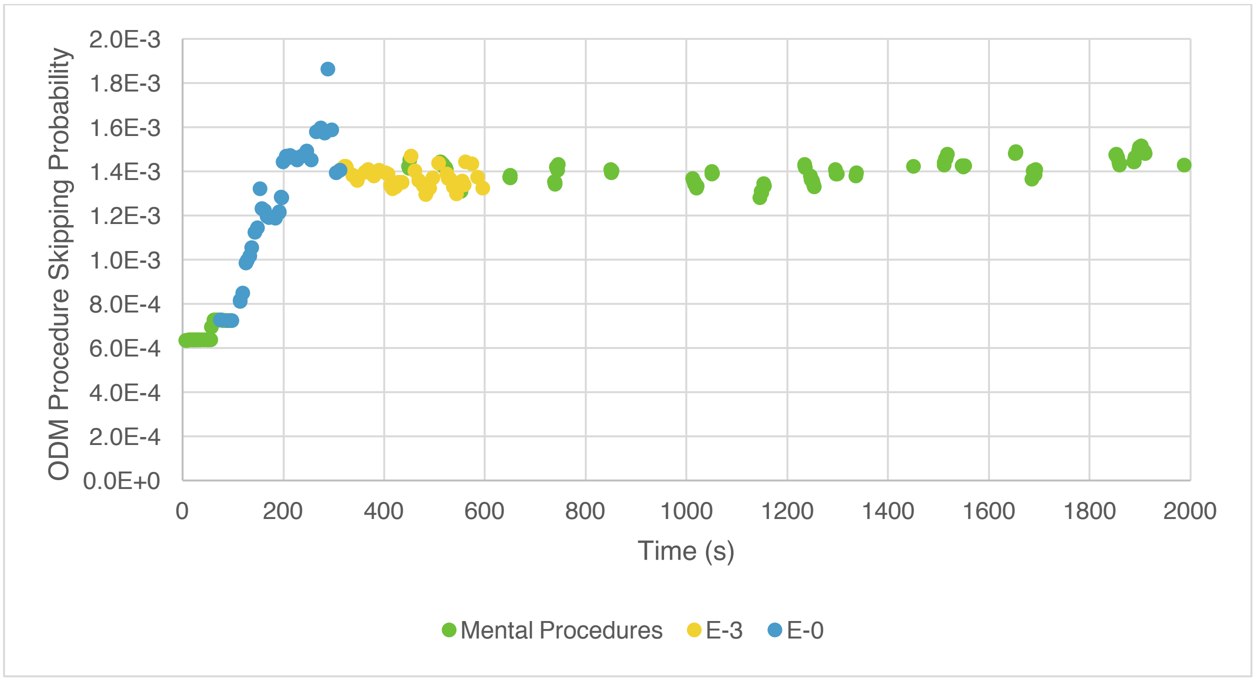


Figure 8-12 The decision-maker's mental procedures, and written procedures E-0 and E-3 step skipping probability

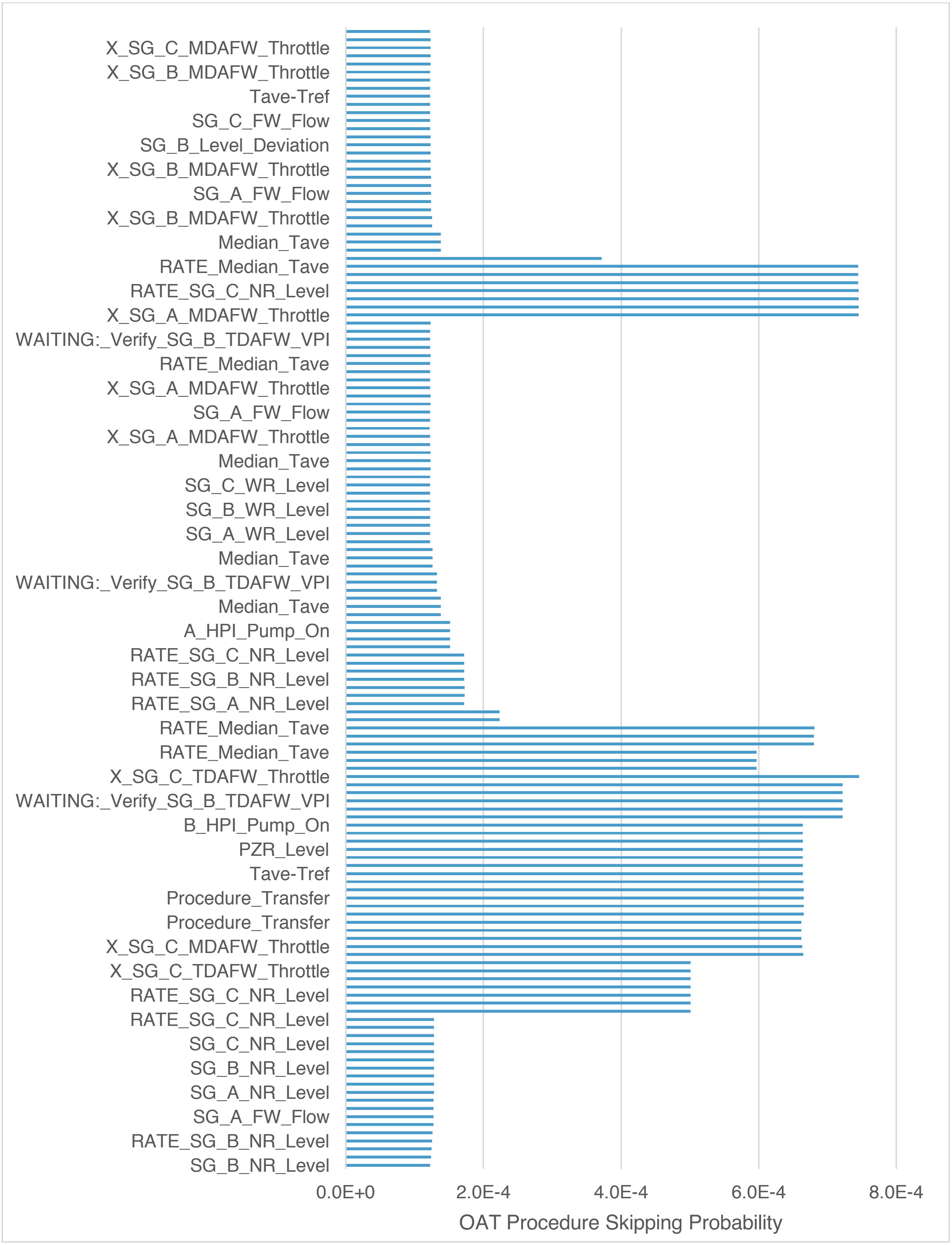


Figure 8-13 Action-taker's sequence of mental procedure events and their skipping probability

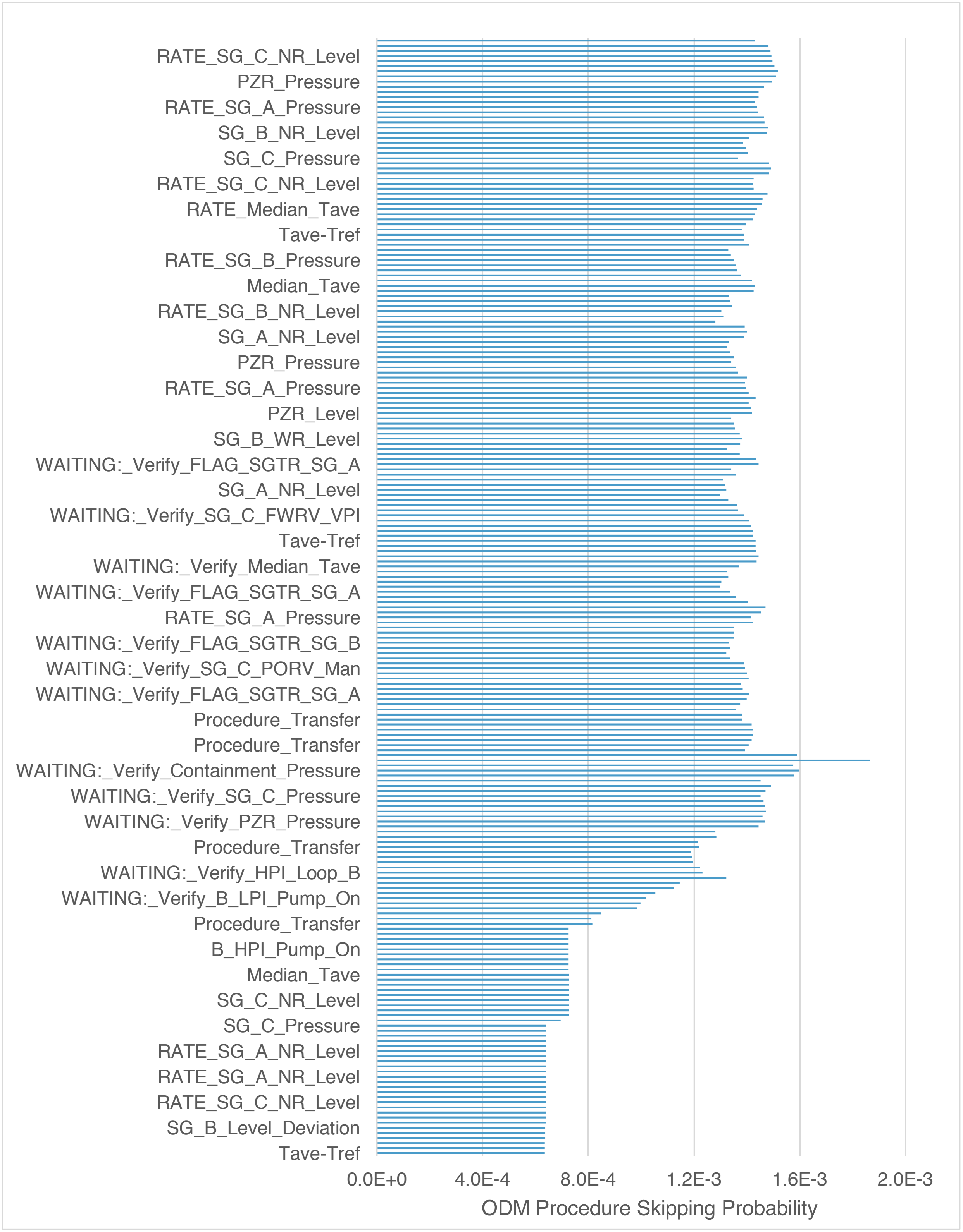


Figure 8-14 Decision-maker's sequence of written and mental procedure events and their skipping probability

In Figure 8-11, the decision-maker's and action-taker's probability of skipping the mental and written procedures at those time steps is shown. The action-taker has only mental procedures available gained through training. Only the decision-maker has access to the procedures. The order of magnitude difference between the action-taker's and decision-maker's probability to skip a procedure is clearly visible. This is mainly because the action-taker does not have to read procedures, but use memorized mental procedures developed through training. Procedures are listed chronologically together with their skipping probability for the action-taker and, respectively, the decision-maker.

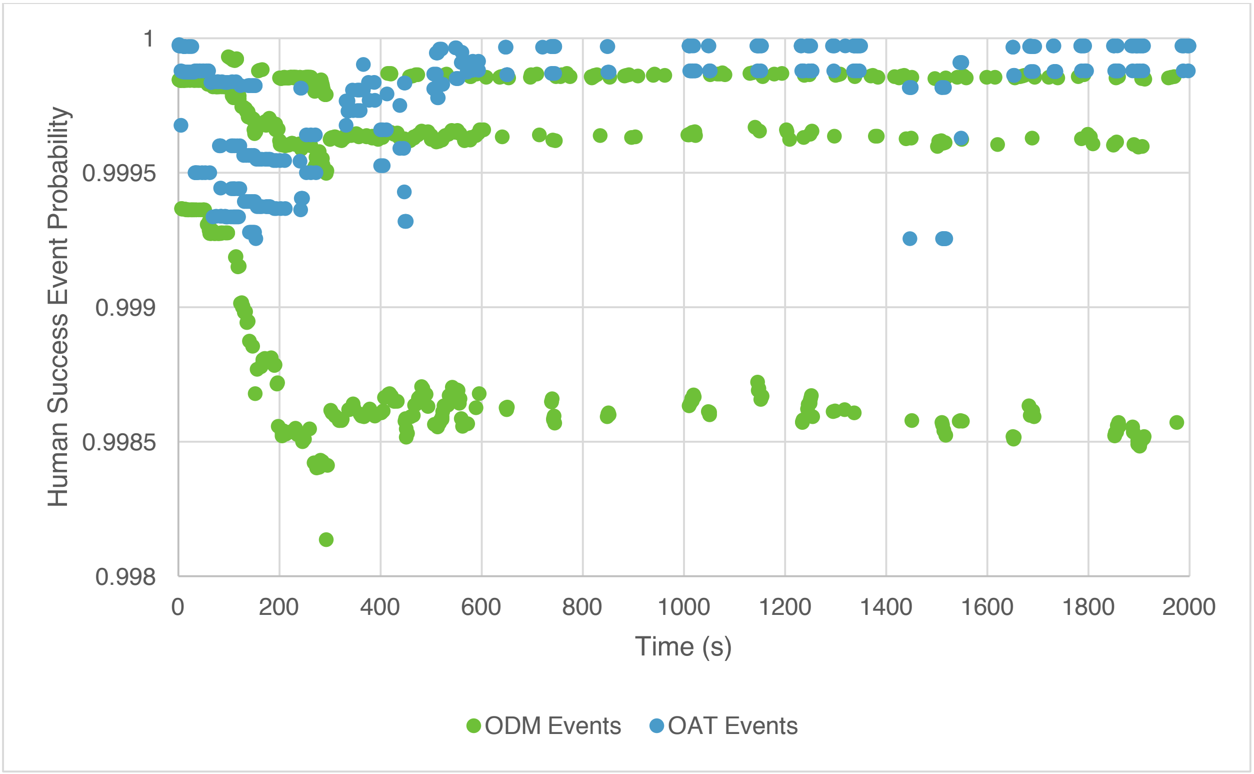


Figure 8-15 Comparison between the decision-maker's and action-taker's success probability of various activities

In Figure 8-12, the decision-maker's probability of skipping the mental procedures, and written procedures E-0 and E-3 are given. This is a good way of visualizing both the influence of context on the probability of skipping a procedure, but also to see when and which procedures where used in diagnosis and recovery. From the change in color, the transfer between E-0 and E-3 happened around 300 seconds into the scenario. In Figures 8-13 and 8-14 the scopes of the mental or written procedures are listed chronologically together with their probability. Note that the same event can have different probabilities depending on the time step and context, for example, the rate of the steam generator A narrow level (i.e. RATE_SG_A_NR_Level).

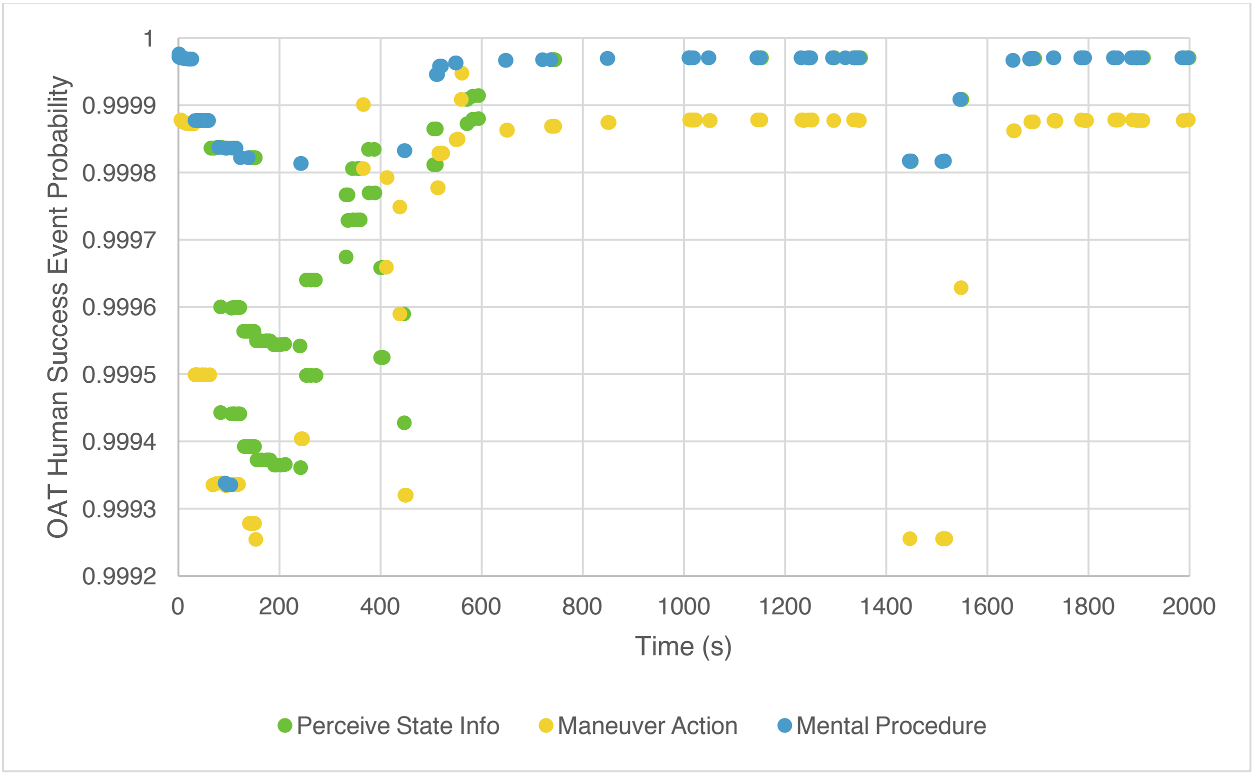


Figure 8-16 Action-taker's success probability for perceiving state information, maneuvering actions and using mental procedures

In Figure 8-15, the probability of success in the all the activities during this sequence by the action-taker and decision-maker are shown together for comparison. Note the changes due to the dynamic environment in which they need to operate. In Figures 8-16 and 8-17, these activities are illustrated further. For the action-taker, the mental procedures have a higher chance of being completed compared to perceiving information when the control room is rather busy with alarms and even compared to the maneuvering of an action because they were developed through training and their recollection requires little effort. Nevertheless, the decision-maker has an even higher change of perceiving new information through the top-down attention mechanism. The procedure interpretation and maneuver action types of activities are more error prone than the action-taker's because of the constant need to be cognitively active for isolating the steam generator.

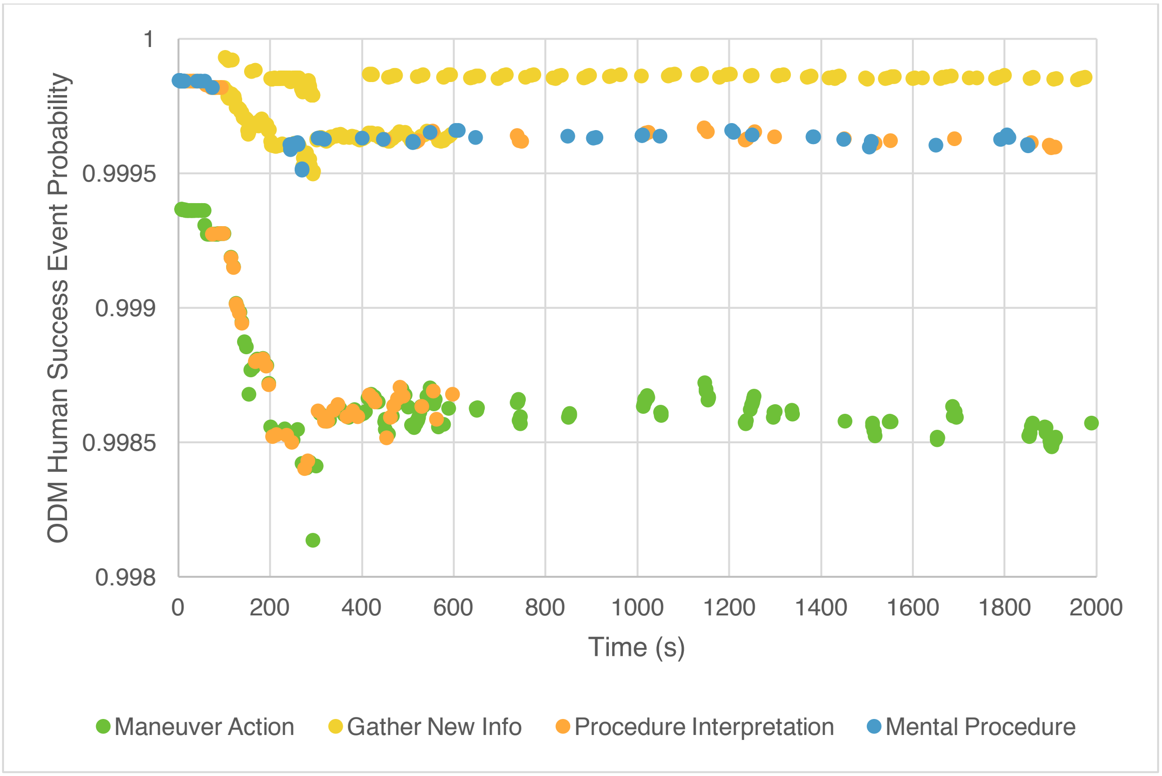


Figure 8-17 Decision-maker's success probability for maneuvering actions, gathering new information, interpreting procedures and using mental procedures

In order to obtain the probability of success for HFE 1B (that is, the crew succeeds to isolate the faulted steam generator) all the decision-maker's and action-taker's event probabilities from the time step the high-pressure injection starts until the crew has isolated steam generator A have been multiplied to obtain: 87.82% with the 5th and 95th percentile bounds 82.69% and, respectively, 89.55%. The confidence bounds have been obtained by running a Monte Carlo simulation with 100,000 samples. Therefore, the probability of HFE 1B (failure of the crew to isolate the faulted steam generator) is 12.18% with the 5th and 95th percentile bounds 10.45% and, respectively, 17.31%. This falls in the band for HFE 1B, which is a reasonable result given that the DBN conditional probability table has been normalized using only HFE 1A (Figure 8-18).

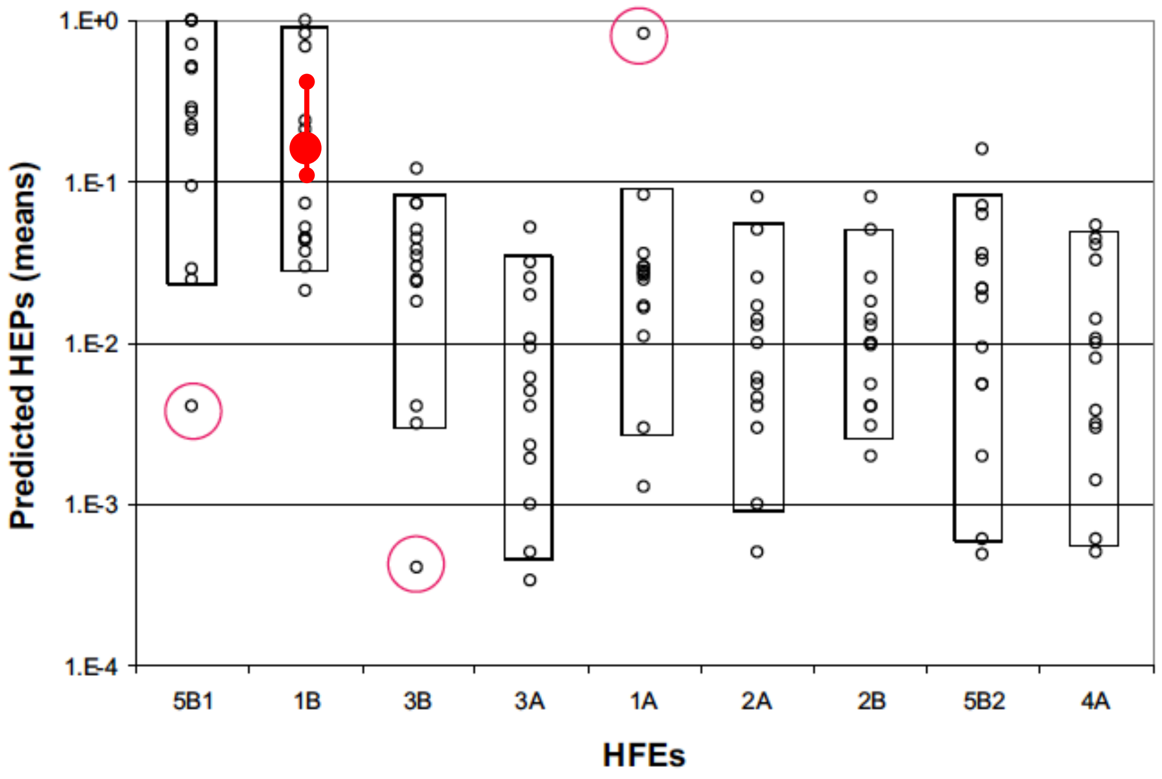


Figure 8-18 HFE 1B prediction of ADS-IDAC

9 Performing dynamic accident sequence precursor analysis with ADS-IDAC

In this chapter, a few details are provided regarding when and why ADS-IDAC is a good simulation tool to perform an ASP analysis. It is intended to be the starting point in performing an ASP analysis. A full guide is provided in Appendix 2. Also, a test case based on of a real accident precursor is given to showcase the new dynamically linked FTs for modeling the impact of support system failures on the frontline systems.

The ASP Program is an important United States NRC programs focused on continuously assessing the risk significance of performance deficiency or degraded conditions. Retrospective event and condition assessment are used to identify and rank the operational events that could potentially lead to accident conditions. It is critical to emphasize that the narrow scope of an ASP analysis is perfectly compatible with the dynamic PRA resource demanding needs since the analysis is intended to be limited only to the operational event of interest with only a few postulated abnormal conditions. It is worth mentioning that the remarks given in the next section are generally applicable any dynamic PRA analysis with ADS-IDAC.

9.1 A few general remarks before proceeding

9.1.1 The recommended analysis team

As with any other PRA or HRA method, a multi-disciplinary team is recommended to be available for the necessary background knowledge of the accident under investigation. However, the actual collection of data and implementation of the model can be performed by any of the team members alone (to be called the "ADS-IDAC analyst").

The recommended analysis team should include people with sufficient knowledge and expertise to supply information about the accident under investigation to build the ADS-IDAC model. Typical skillsets and specialists include:

- a PRA analyst,
- an HRA analyst,
- a thermal-hydraulics engineer providing the RELAP5 assistance,
- a reactor operations trainer, and
- a reactor operator.

9.1.2 Necessary information for performing the analysis with ADS-IDAC

The multi-disciplinary team of experts should provide knowledge on a wide range of topics necessary for performing the analysis by the ADS-IDAC analysist. It is the responsibility of the ADS-IDAC analyst to understand the ADS-IDAC methodology for collecting the appropriate information, perform expert elicitation, and training the other team members on ADS-IDAC if needed. In general, the following types of information are needed to perform an ASP analysis with ADS-IDAC:

- existing plant-specific PRA;
- underlying ADS-IDAC methodology;
- NPP behavior (including both frontline and support systems);
- operator training programs;
- NPP procedures and guidelines; and
- NPP operational history.

The easiest was to perform an analysis with ADS-IDAC is to check existing simulation models for the design and scenario under investigation in the ADS-IDAC installation folder under ‘testcases’. If such simulation models are available, minimal changes are required and the analyst can proceed directly to add new conditional events. By running the simulations, the analysts can immediately assess how the added conditional events or NPP conditions contribute to the overall NPP and crew performance and, ultimately, to the core damage frequency.

Until a comprehensive database of ADS-IDAC simulation models becomes available, it is likely that the analysts would need to create a new simulation model that matches their needs. The step-by-step guide found in the next section is intended to help the analysts understand how such simulations models could be created and the exact information needed to complete it. It is expected that this guide will be followed sequentially as some model elements may depend or other elements that need to be defined in advance (e.g. conditional failure events cannot be defined for components that have not been previously created in the control panel).

9.1.3 Advantages and limitations of using ADS-IDAC

The ADS-IDAC offers several key advantages over the conventional FT and ET PRA methods:

- it can capture the impact of event sequence timing,
- it provides a better representation of thermal-hydraulic success criteria,
- it permits more detailed and realistic modeling of operator response,
- the complexity of enumerating scenarios is delegated to scenario generating algorithms,
- it reduces analyst-to-analyst variability of the results; and
- it allows heterogeneous models of various phenomena to be devolved and used at different levels of detail (simulation tracking can provide desired information on nature of scenarios – "white box" simulation)

Nonetheless, ADS-IDAC shares the following limitations with all discrete dynamic PRA models:

- physical models can be resource intensive to develop and difficult to validate (particularly for rare events);

- obtaining a complete risk profile (i.e. ensuring that a complete solution space is examined and representative samples are chosen) requires further research;
- efficient methods for uncertainty analysis do not exist as certain types of uncertainty and variability can alter the structure of risk scenarios as they evolve over the time; and
- possible "scenario space explosion" noting that "smart" algorithms have been explored that produce dominant risk scenarios at reasonable simulation time.

9.1.4 When to use ADS-IDAC

Given ADS-IDAC benefits and limitations, it is reasonable to ask "when should one use ADS-IDAC?" Considering that ADS-IDAC has thus far only been used for validation and testing purposes, this question does not have a straight forward answer. For existing ADS-IDAC models or separate RELAP5 physical models, it should be apparent that ADS-IDAC is a very powerful tool given the advantages listed in the previous section.

Even if new ADS-IDAC models are needed, there are certain situations when the effort to build the model will be worthwhile. In general, these cases have some of the following characteristics that are necessary to be captured in the analysis:

- control loops;
- complex hardware – human interactions;
- multiple NPP conditions or aleatory influences that may affect the operator behavior;
- operator-to-operator variability; and
- complex design or procedural changes that may affect the operator behavior.

9.1.5 Creating a new ADS-IDAC model

If an ADS-IDAC model is not readily available for the application under study, a new model would have to be created starting from a RELAP5 model. Appendix 2 contains a user guide for creating such a model.

Moreover, the ASP analysis sequence of interest can be used as input for the dynamic PRA model to constrain the simulation to the solution space of interest. To be more precise, in ADS-IDAC, by simulating the plant and crew response models, and using the generic branching rules, the initial sequence of events is expanded into a DDET capturing both the initial sequence and various other sequences with different timing of failures, order of failures, degrees of component degradation, time and degree of recovery, human actions, human decisions or physical variable thresholds. This can be encapsulated in the cut set diffraction phenomenon graphically shown in Figure 9-1.

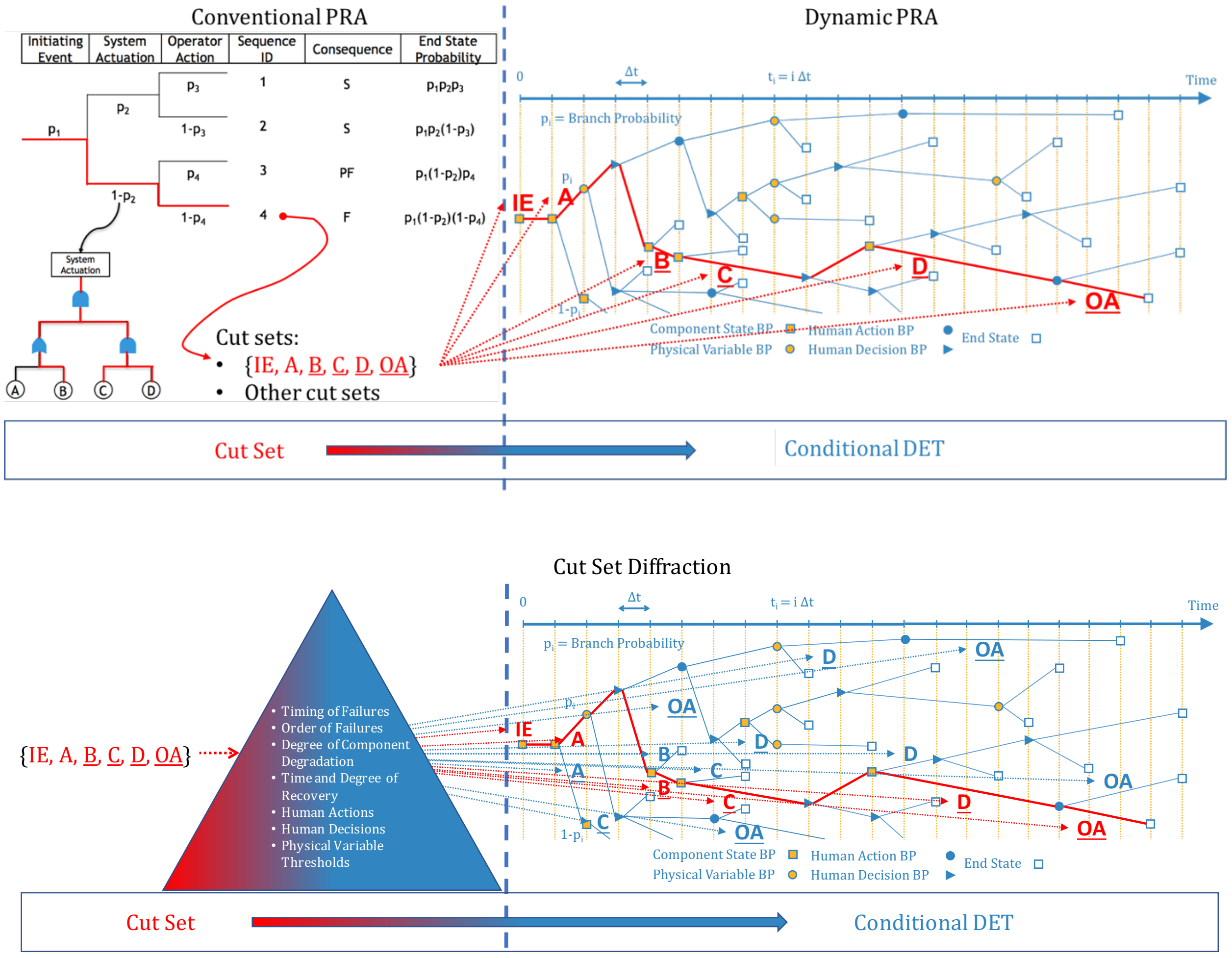


Figure 9-1 Cut set diffraction phenomenon

9.1.6 Analysis of results

Given the availability of a complete ADS-IDAC model, the scope of the ADS-IDAC analysis is defined by activating branching rules and setting sequence termination conditions. Multiple runs of the simulation model can be performed where the input conditions can be varied. This effectively allows the analyst to investigate multiple “what if?” scenarios and compare the results between similar conditions with slightly altered precursors.

9.2 Test case for accident sequence precursor analysis of a PWR NPP trip with loss of reactor coolant pump seal injection and cooling due to electrical fault

A test case application of the dynamically liked FTs in the DDET generated by ADS-IDAC considered in this research was inspired by the March 28, 2010 H. B. Robinson NPP trip due to an electrical failure complicated by concurrent equipment failures and inappropriate crew diagnosis and control of the NPP.

9.2.1 Electrical Fault Causes Fire and Subsequent Reactor Trip with a Loss of Reactor Coolant Pump Seal Injection and Cooling

At 18:52 EDT on March 28, 2010, with the H. B. Robinson Steam Electric Plant, Unit No. 2, was operating in Mode 1 at approximately 99.5% power. A feeder cable failure to 4kV non-vital Bus 5 caused an arc flash and fire. The 4kV Bus 5 failed to isolate from non-vital 4kV Bus 4 due to a failure of the circuit breaker 52/24 to open, which resulted in reduced power to Reactor Coolant Pump (RCP) B and a subsequent reactor trip on Reactor Coolant System (RCS) loop low flow. After the reactor trip, an automatic safety injection occurred due to RCS cooldown. Plant response was complicated by equipment malfunctions and failure of the operating crew to diagnose the NPP’s condition and properly control the NPP. A second fire occurred during a reset of an electrical distribution system control relay, but it was not considered in this analysis. Additional detailed information is available in Sanders (2010).

The H. B. Robinson Westinghouse PWR electrical system is designed to diversely and independently supply power from various sources. Two power systems make up the whole NPP electrical system: the main power system (the main generator with its necessary controls for off-site distribution to the grid), and the auxiliary power system (takes energy from the off-site grid, which is referred to as 'system auxiliary power' or from the main power system during operation which is referred to as 'unit auxiliary power.')

During the startup procedure, all the power is provided by the system auxiliary power through one or more redundant system auxiliary transformers from independent buses of the off-site distribution grid. Due to voltage mismatch, the unit auxiliary power employs the unit auxiliary transformer to convert supplied voltage from the main generator to the NPP electrical distribution voltage.

During at-power operation, the NPP distribution buses are partly supplied by the main generator and partly by the system auxiliary transformers. Given the simple arrangement of buses, the electrical system requires minimal switching to restore power to a bus in case its normal power supply is faulted. A simplified arrangement of the H. B. Robinson Steam Electric Plant, Unit No. 2's electrical system is shown in Figure 9-2. For increased safety and reliability during loss of power events, the transfer of energy supply is automatic and the NPP buses are powered from two diverse sources.

9.2.2 Precursor Analysis

In the final precursor analysis of H.B. Robinson Electrical Fault Causes Fire and Subsequent Reactor Trip with a Loss of RCP Seal Injection and Cooling released by the United States NRC in 2010, the Robinson event ASP analysis was modeled as a loss of main feedwater (LOMFW) transient initiating event with additional failures to realistically model the loss of the RCP seal injection and cooling (LOSC), and subsequent LOCA. The analysis was performed with conventional PRA tools.

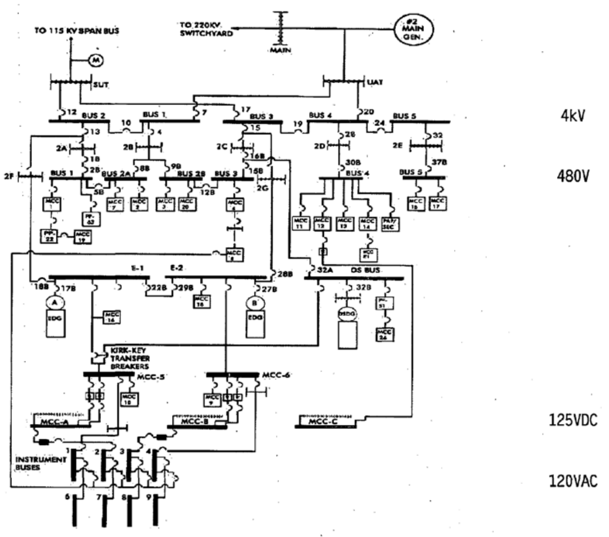


Figure 9-2 Robinson PWR NPP simplified electrical system (Sanders, 2010)

The events and important component failures in the most dominant accident sequence (contributing 68% of the total internal events conditional core damage probability), LOMFW Sequence 02-14-04, are:

- Loss of MFW transient occurs,
- Reactor trip succeeds,
- Auxiliary feedwater (AFW) succeeds,
- Power-operated relief valves (PORVs) successfully close,
- Loss of RCP seal cooling/injection occurs,
- Operators fail to trip the RCPs,
- Subsequent RCP seal LOCA occurs,

- Safety injection (SI) succeeds,
- Operators successfully cooldown/ depressurize the RCS,
- Operators fail to initiate shutdown cooling (SDC) mode of the residual heat removal (RHR) system, and
- High-/low-pressure recirculation fails.

The following events were of key interest for the test case that was set up for ADS-IDAC to include support systems failures:

- The Unit Auxiliary Transformer (UAT) failed because of an overload condition caused by the ground fault on Bus 5. This led to the automatic transfer of 4kV Buses 1, 4, and 5 to the Startup Transformer (SUT).
- As designed, the reactor automatically tripped due to low reactor coolant flow triggered by an under-voltage condition on Bus 4, and, subsequently leading to a decrease in RCP B speed. The MFW was isolated when the SI signal occurred and AFW initiated to provide makeup to the steam generators.
- RCP seal cooling via Component Cooling Water (CCW) was unavailable as the Flow Control Valve (FCV) 626 was closed. This was caused by an inaccurate high-flow signal when the flow sensor lost power via Instrument Bus 4 due to the momentary loss of power to vital Bus E2. The recovery of seal cooling by the operators was delayed because they initially did not use RCP Thermal Barrier Cooling Water Low Flow Annunciator procedure directing them to verify the FCV-626 position and reopen it if closed.

- RCP seal injection was assumed either unavailable or inadequate to fulfill its safety function because the opening of the Chemical and Volume Control (CVC) Valve 310A caused the switch of flow from the RCP seals to the RCS. With seal cooling unavailable until the operators reopen FCV-626, the RCP seal LOCA occurs. The charging pump suction source failed to automatically switch-over from the Volume Control Tank (VCT) to the Refueling Water Storage Tank (RWST) on low VCT level.
- Several electrical systems were unavailable caused by the transient and electrical faults: off-site power was lost to vital Bus E2, Non-vital Bus 5 was unavailable due to damage from the electrical fault, and Non-vital Bus 4 was unavailable due to the electrical fault on Bus 5 and the failure of the bus tie-breaker (Breaker 52/24) to open.

It is worth mentioning here that a subsequent rapid cooldown of the RCS occurred that was terminated by the automatic closing of the MSIVs, and after the NPP had reached a safe shut down state, a second fire was initiated by the operators when they re-energized the initial fault trying to reset the generator lockout relays. If additional breakers had faulted, a site-wide loss of off-site power would have been possible.

9.2.3 ADS-IDAC simulation model

This test case was set up to showcase the application of the newly implemented capability to include support system failures that have an impact on the frontline system performance. Out of the sequence of events captured in the Final Precursor Analysis and mentioned in the previous section, of interest to the ADS-IDAC analysis focuses on all the events up to the initiation of the RCP seal LOCA following LOSC.

A generic three-loop, PWR NPP RELAP5 model was used as the starting point to set up the ADS-IDAC model. This thermal-hydraulic model currently has 75 controls, 180 indicators, and 70 alarms, which gave us a good foundation to realistically model the H. B. Robinson electrical fault precursor scenario.

The simulation was initiated by the loss of MFW, followed by automatic reactor trip with AFW available and PORVs closed.

Concurrently, a series of faults are modeled to lead to a LOSC event. The LOSC event is dynamically linked to the Modified Robinson Loss of Seal Cooling FT (Figure 9-3). In the Modified Robinson Loss of Seal Cooling FT, two transfer gates are used to include the Modified Robinson RCP B Seal Injection FT (Figure 9-4) and Modified Robinson Emergency Bus E2 FT (Figure 9-5).

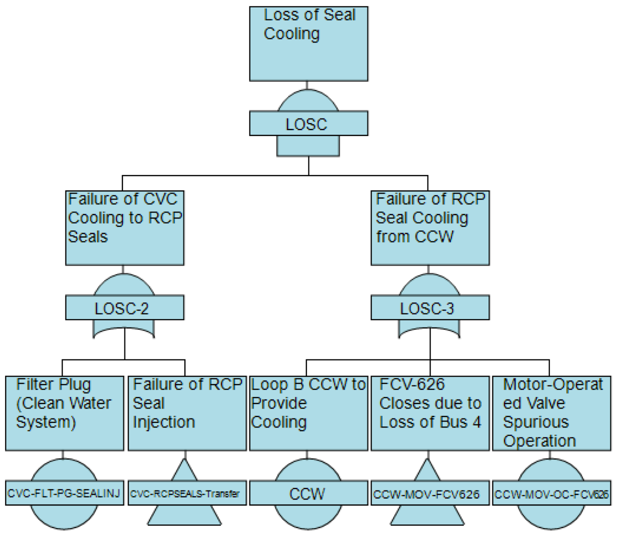

Loss of Seal Cooling
LOSC
Failure of CVC Cooling to RCP Seals
LOSC-2
Failure of RCP Seal Cooling from CCW
LOSC-3
Filter Plug (Clean Water System)
CVC-FLT-PG-SEALINJ
Failure of RCP Seal Injection
CVC-RCPSEALS-Transfer
Loop B CCW to Provide Cooling
CCW
FCV-626 Closes due to Loss of Bus 4
CCW-MOV-FCV626
Motor-Operated Valve Spurious Operation
CCW-MOV-OC-FCV626

Figure 9-3 Modified Robinson Loss of Seal Cooling FT

The initial basic event probabilities are taken from the Final Precursor Analysis of United States NRC 2010, however their values do not have a significant importance for this test case as more important to this analysis are the qualitative results. During the simulation, as in conventional FTs, a component failure is modeled as a basic event by setting its failure probability to 1.0.

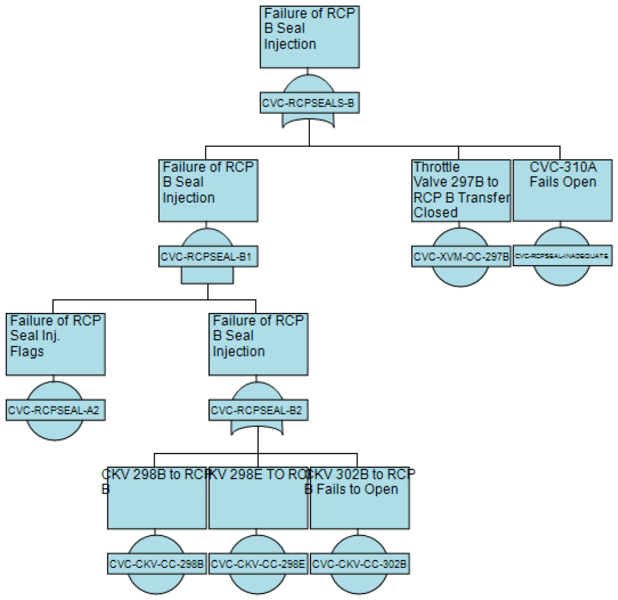

Failure of RCP B Seal Injection
CVC-RCPSEALS-B
Failure of RCP B Seal Injection
CVC-RCPSEAL-B1
Throttle Valve 297B to RCP B Transfer Closed
CVC-XVM-OC-297B
CVC-310A Fails Open
CVC-RCPSEAL-INADEQUATE
Failure of RCP Seal Inj. Flags
CVC-RCPSEAL-A2
Failure of RCP B Seal Injection
CVC-RCPSEAL-B2
CKV 298B to RCP B
CKV 298E TO RCP
CKV 302B to RCP B Fails to Open
CVC-CKV-CC-298B
CVC-CKV-CC-298E
CVC-CKV-CC-302B

Figure 9-4 Modified Robinson RCP B Seal Injection FT

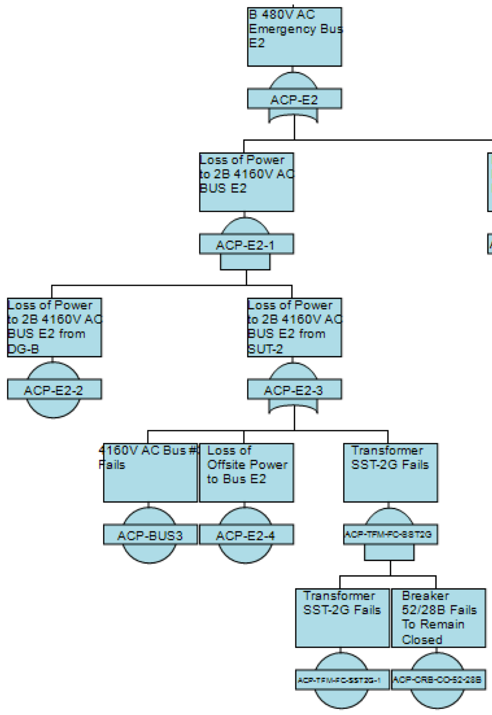

B 480V AC Emergency Bus E2
ACP-E2
Loss of Power to 2B 4160V AC BUS E2
ACP-E2-1
Division 2B AC Power 4160V BUS E2
ACP-BAC-LP-E2
Loss of Power to 2B 4160V AC BUS E2 from DG-B
ACP-E2-2
Loss of Power to 2B 4160V AC BUS E2 from SUT-2
ACP-E2-3
4160V AC Bus #3 Fails
ACP-BUS3
Loss of Offsite Power to Bus E2
ACP-E2-4
Transformer SST-2G Fails
ACP-TFM-FC-SST2G
Transformer SST-2G Fails
ACP-TFM-FC-SST2G-1
Breaker 52/28B Fails To Remain Closed
ACP-CRB-CO-52-28B

Figure 9-5 Modified Robinson Emergency Bus E2 FT

To realistically capture the sequence of events that occurred at H. B. Robinson after the initial electrical failure and to explore other possible outcomes, various conditions were modeled as system reliability failures on demand, during operation or at specified times during the simulation:

- Off-site power was lost to vital Bus E2.
- Emergency Diesel Generator (EDG) B fails to start on demand.
- CVC-310A valve failed open because of a loss of instrument air.

The basic events and top events of the FTs are also included in the procedures to allow the operators to verify and recover certain faults or model operator failures related to these faults. For example, recovery of off-site power due to vital Bus E2 could be achieved through the non-vital Bus 3. Also, using the procedures the operators can verify and reopen FCV-626.

9.2.4 Results

The DDET generated with ADS-IDAC is shown in Figure 9-6. The given DDET shows only the branching events and 15 sequences, as the full DDET with all the non-branching events could not be graphically included in the paper.

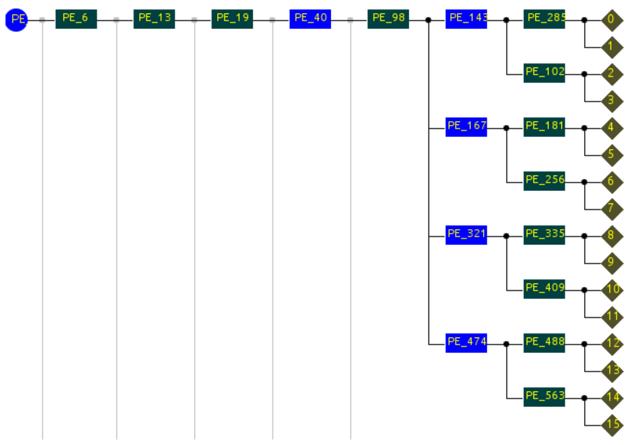


Figure 9-6 A Portion of ADS Generated DDET

Aside from the branching points based on the system failures, events related to actions taken by the operators during the procedure following strategy are also included.

More than one failure is allowed at each time step, therefore there are cases where more than two branches are created at that specific time. For example, PE_143, PE_167, PE_321, and PE_474 were generated for combinations of failure or success of two events: loss of off-site power to vital Bus E2 and EDG B fails on demand.

The individual branching events and underlying failure conditions may not be risk significant, however their combination in the DDET may create new contexts that can lead to new unexplored accident conditions.

In Figure 9-7, the pressurizer pressure time evolution is given for the sequences with end states (ES) 4, and respectively, 5. The differences between these sequences is the operator's manual actuation of the emergency core cooling system (ECCS) based on the action step in the EOPs.

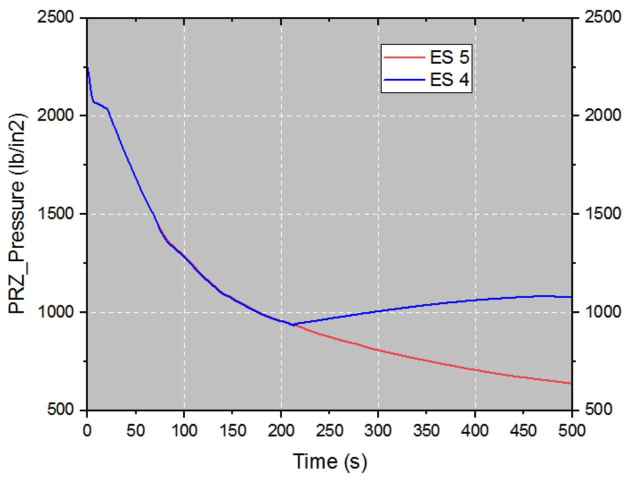


Figure 9-7 Pressurizer pressure vs time for ES 4 and ES 5

It should be noted that these results should be viewed as simply a way to illustrate new dynamic linking of FTs and branching capabilities of the ADS-IDAC's platform by simulating an actual scenario and not a full and realistic replication of the results in Final Precursor Analysis of United States NRC.

10 Summary and conclusions

The main goal of this research was to develop the necessary features to augment the ADS-IDAC simulation engine for more practical and realistic applications (especially event assessments) and as a tool to analyze highly dynamic and complex accident scenarios in support of conventional PRAs. This was achieved by the following:

1. Integrating support system FT models into the dynamic simulation runs and tracing their impact on the frontline systems by developing and implementing an algorithm for incorporating binary logic of system failures into the dynamic branching rules of the dynamic ETs.

2. A simplified version of the hybrid response mode was developed and implemented to replicate a knowledge-based procedure-following response mode.

3. A set of comprehensive quantification rules to enable dynamic calculation of branch probabilities and complete risk scenario probabilities was developed and implemented using a DBN. The HFE dependencies were explicitly accounted for through the shared PSFs by adapting the BBN model of PSFs developed in the Phoenix method to the dynamic environment of ADS-IDAC.

4. Selecting appropriate probability distributions for the HFE and creating capability to select various probability distributions for the hardware failure events, and implementing algorithms to propagate the uncertainties through the DDET.

5. Developing graphical software enhancements to support the current and previous tasks to the ADS-IDAC graphical user interface.

6. Describing and demonstrating why, when, and how ADS-IDAC can be used for ASP analysis studies.

11 Suggestions for future work

- For the International HRA Empirical Study SGTR complex scenario, all the predicted sequences to fail HFE 1B can be compiled and their relative importance to the total probability can be assessed.
- In the HCL library, uncertainty propagation through BBNs can be implemented to complete the uncertainty analysis to all three layers: ESDs, FTs, and BBNs. In turn, this would allow the uncertainty analysis in ADS-IDAC to be expanded to the 'PSFs – crew failure modes DBN' and eventually propagate the uncertainties while the DDET is generated.
- Adaptive sampling techniques and surrogate models for uncertainty propagation can be implemented to reduce the computational costs incurred by the thermal-hydraulic code RELAP5.
- Use the SACADA database (Chang, 2014) for human reliability and human performance to update the parameters used in the DBN.
- ADS-IDAC has matured enough to be ready for an integration into a hybrid, conventional and dynamic, framework for full scope NPP risk assessments complete with scenarios, likelihood, and consequences. The full scope NPP PRAs developed by the industry could be a valuable reference both in terms of availability of data, and as a general theory of risk assessment.

Appendix 1: Guide for building the simulation engine executable from the source code

This guide is also included and updated regularly in the source code repository.

The ADS-IDAC simulation engine has been tested on a variety of platforms:

- Ubuntu 16
- MacOS 10.12
- Windows 10

At the time of writing this thesis, each platform requires CMake v 3.8.0 or newer, and the latest Intel C++ and Fortran Compilers. Moreover, GNU-compatible Make, Xcode Developer Tools v8, and Microsoft Visual C++ v14 are required on Ubuntu, macOS, and, respectively, Windows.

ADS-IDAC comes with a CMake build script (CMakeLists.txt) that can be used on a wide range of platforms ("C" stands for cross-platform). If you don't have CMake installed already, you can download it for free from http://www.cmake.org/. CMake works by generating native makefiles or build projects that can be used in the compiler environment of your choice.

Also, before building ADS-IDAC, the boost libraries need to be installed. Please go to http://www.boost.org and follow the instructions under "Getting Started" for your environment.

When building ADS-IDAC as a standalone project, the typical workflow in terminal or command prompt starts with:

- mkdir build   # Create a directory to hold the build output.
- cd build

- cmake ../src -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_COMPILER=icpc –DCMAKE_Fortran_COMPILER=ifort # Generate native build scripts.

Note: The supported build types are: Release and Debug. Also, on Windows replace icpc with icl.

If you are on an Ubuntu system, you should now see a Makefile in the current directory. Just type 'make' in terminal to build ADS-IDAC.

On Mac OS X with Xcode installed, a .xcodeproj file will be generated. Open the .xcodeprof file in Xcode and build the executable.

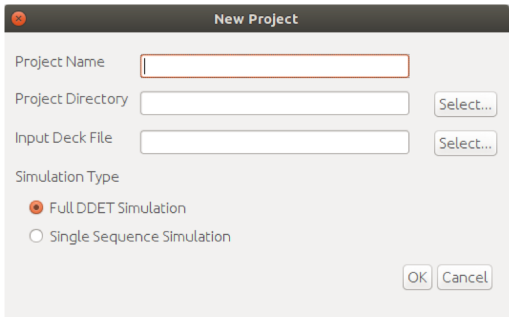
If you use Windows and have Visual Studio installed, a .sln file and several .vcproj files will be created. You can then build them using Visual Studio by opening the .sln file and building the solution to create the ADS-IDAC executable.

Appendix 2: Step-by-step guide for creating a new ADS-IDAC model

This step-by-step guide was written starting from the ADS-IDAC 2.0 (Coyne, 2009) and ADS-IDAC 3.0 (Yuandan, 2013) simulation engine developmental work.

Step 1: Create New Project

From the menu bar or graphics toolbar select "New Project". A pop-up window will show up requiring the following information:



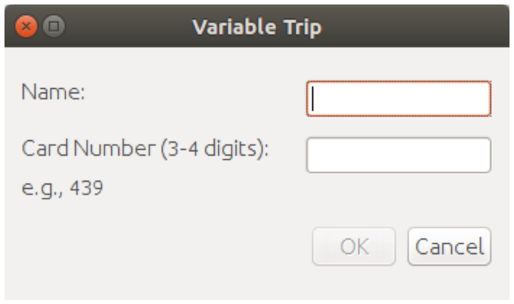
"Project Name" - type a name for the project;

"Project Directory" - select a directory to save it into;

"Input Deck File" - select a RELAP5 thermal-hydraulic plant model (i.e. *.i file); and

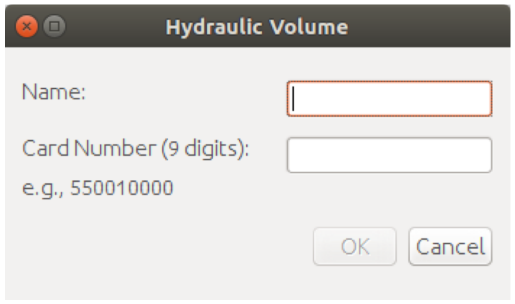
"Simulation Type" - select the simulation type.

This will enable access to the other tabs in the application.



Step 2: Create RELAP5 communication channels

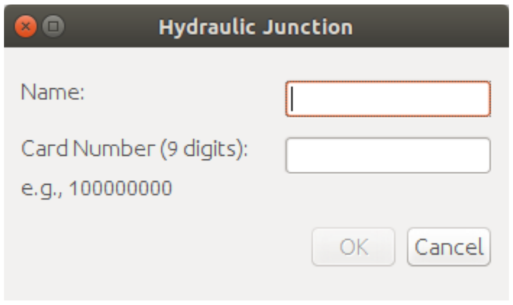
In the "Plant Model -> Frontline System Thermal-Hydraulic Model" tab, the RELAP5 communication channels are created. They act information bridges between the thermal-hydraulic RELAP5 model and the ADS-IDAC control panel.



By clicking on the "Hydraulic Volume Insert" icon from its toolbar, a pop-up window will show up requiring the following information:

"Name" - these channels are denoted by the prefix "HV_" and are used to establish a communication link to RELAP volumetric elements (e.g. temperature, pressure, void fraction, vapor quality); and

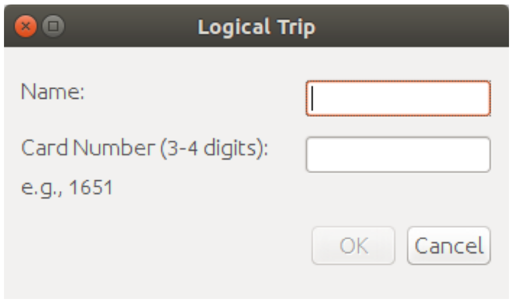
"Card Number" - it must be a valid hydraulic volume card number specified in the RELAP5 input deck.



By clicking on the "Hydraulic Junction Insert" icon from its toolbar, a pop-up window will show up requiring the following information:

"Name" - these channels are denoted by the prefix "HJ_" and are used to establish a communication link to RELAP junction elements (e.g mass flow rate); and

"Card Number" - it must be a valid hydraulic junction card number specified in the RELAP5 input deck.



By clicking on the "Variable Trips Insert" icon from its toolbar, a pop-up window will show up requiring the following information:

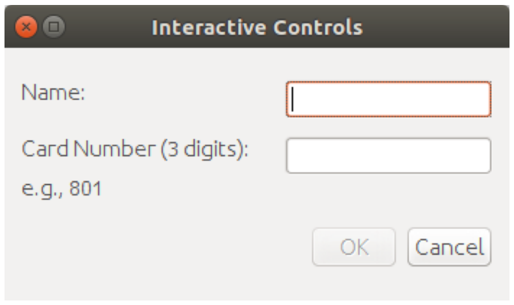
"Name" - these channels are denoted by the prefix "VT_" and are used to establish a communication link to RELAP variable trips elements (e.g alarms and component states).

"Card Number" - it must be a valid variable trip card number specified in the RELAP5 input deck.

By clicking on the "Logical Trips Insert" icon from its toolbar, a pop-up window will show up requiring the following information:

"Name" - these channels are denoted by the prefix "LT_" and are used to establish a communication link to RELAP logical trips elements (e.g alarms and component states); and

"Card Number" - it must be a valid logical trip card number specified in the RELAP5 input deck.


Interactive Controls

Name:

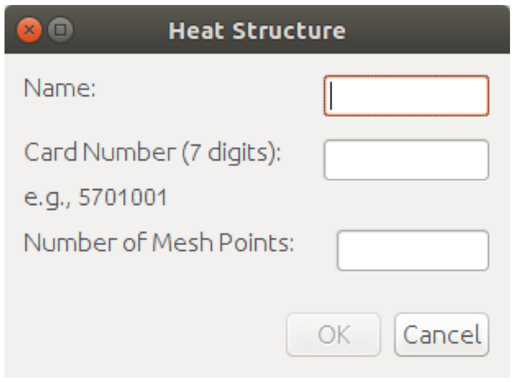
Card Number (3 digits):
e.g., 801

By clicking on the "Interactive Controls Insert" icon from its toolbar, a pop-up window will show up requiring the following information:

"Name" - these channels are denoted by the prefix "IC_" and are used to establish a communication link to RELAP interactive controls elements (e.g pumps, valves, actuation switches); and

"Card Number" - it must be a valid interactive control card number specified in the RELAP5 input deck.

By clicking on "Heat Structures Insert" icon from its toolbar, a pop-up window will show up requiring the following information:


Heat Structure

Name:

Card Number (7 digits):
e.g., 5701001

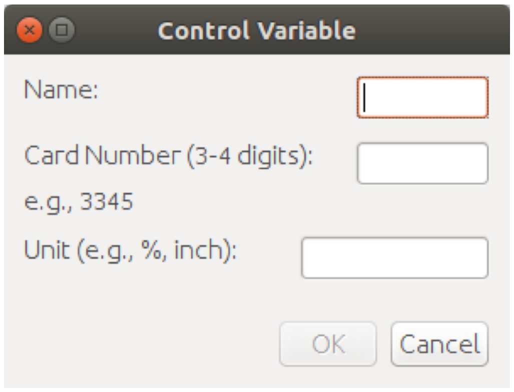
Number of Mesh Points:

"Name" - these channels are denoted by the prefix "HS_" and are used to establish a communication link to RELAP heat structure elements (e.g fuel pins, reactor vessel walls, steam generator shells, pressurizer heaters);

"Card Number" - it must be a valid heat structure card number specified in the RELAP5 input deck; and

"Number of Mesh Points" - it must be a valid heat structure radial mesh point specified in the RELAP5 input deck.

By clicking on "Control Variables Insert" icon from its toolbar, a pop-up window will show up requiring the following information:



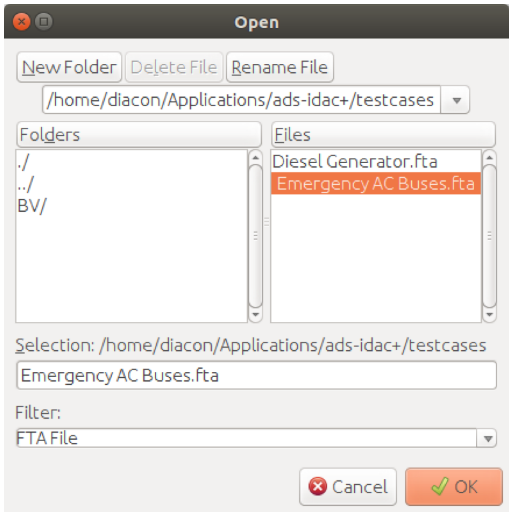
"Name" - These channels are denoted by the prefix "CV_" and are used to establish a communication link to RELAP control variable elements (e.g automatic control systems, derived quantities: average reactor coolant temperature, subcooling margin);

"Card Number" - it must be a valid control variable card number specified in the RELAP5 input deck; and

"Unit" - it must correspond with the specified control variable card measurement system defined in the RELAP5 input deck.

Step 3: Import Support Systems Fault Trees

In the "Plant Model -> Support Systems Fault Trees" tab, the *.fta fault tree files are imported. These are *.xml formatted files that can be generated with the Trilith software.



/home/diacon/Applications/ads-idac+/testcases

./
../
BV/

Diesel Generator.fta
Emergency AC Buses.fta

Selection: /home/diacon/Applications/ads-idac+/testcases
Emergency AC Buses.fta

FTA File

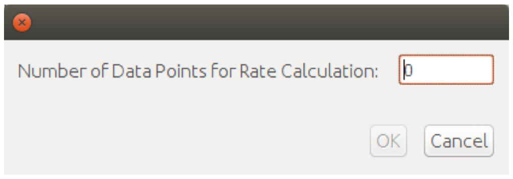
By clicking on the "Import" icon, a pop-up window will show up requiring the selection of an *.fta file.

Step 4: Create Plant Control Room Parameters and States

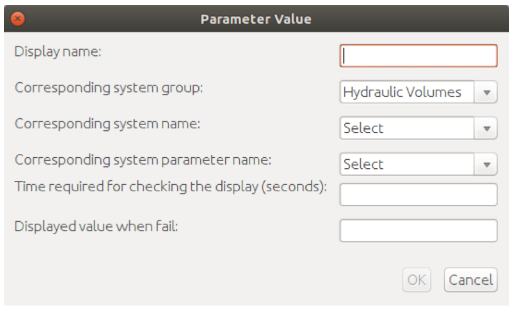
In the "Plant Model -> Plant Control Room -> Parameters & States" tab, the control panel parameters and states are created. All the parameters and states information perceived by the crew need to be displayed on the control panel.

The control panel parameter indicators can display both the value of an indicator and rate of change of the target parameter. The rate of change of a parameter can be used to provide a trend display for use by the crew.

By clicking on "Number of Data Points for Rate Calculation Edit" icon, a pop-up window will show up requiring an integer number.


Number of Data Points for Rate Calculation: 0

By clicking on the "Parameter Value Insert" icon, a pop-up window will show up requiring the following information:


Parameter Value

Display name:
Corresponding system group: Hydraulic Volumes
Corresponding system name: Select
Corresponding system parameter name: Select
Time required for checking the display (seconds):
Displayed value when fail:

"Display Name" - descriptive name for the control panel parameter value indicator. Underscore "_" should be used instead of spaces. Any control panel parameter indicator can be used to provide trend information by adding the prefix "RATE_" to the Display Name.

"Corresponding System Group" - select one of the available options.

"Corresponding System Name" - select one of the available options.

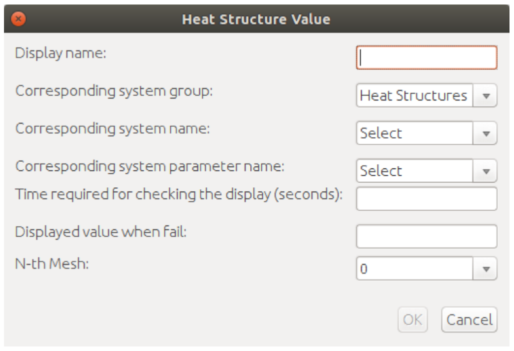
"Corresponding System Parameter Type" - select one of the available options.

"Time Required for Checking the Display (seconds)" - used to specify the time required for the operator to read a control panel indicator.

"Displayed Value When Fail" - used to specify the value indicated by the associated indicator after a failure.

By clicking on the "Heat Structure Value Insert" icon, a pop-up window will show up requiring the following information:

“Display Name” - descriptive name for the control panel heat structure value indicator. Underscore “_” should be used instead of spaces. Any control panel parameter indicator can be used to provide trend information by adding the prefix “RATE_” to the Display Name.


Heat Structure Value
Display name:
Corresponding system group:
Heat Structures
Corresponding system name:
Select
Corresponding system parameter name:
Select
Time required for checking the display (seconds):
Displayed value when fail:
N-th Mesh:
0
OK
Cancel

“Corresponding System Group” - select one of the available options.

“Corresponding System Name” - select one of the available options.

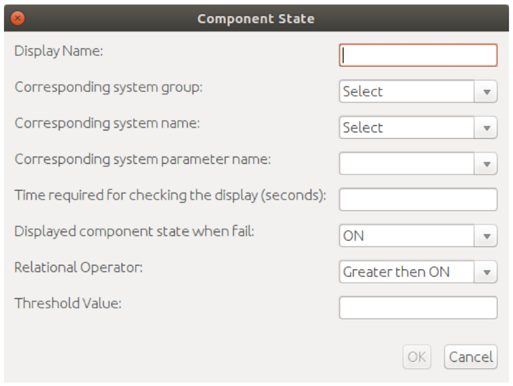
“Corresponding System Parameter Type” - select one of the available options.

“Time Required for Checking the Display (seconds)” - used to specify the time required for the operator to read a control panel indicator.

“Displayed Value When Fail” - used to specify the value indicated by the associated indicator after a failure.

“N-th Mesh” - select one of the available options.

By clicking on the “Component State Insert” icon, a pop-up window will show up requiring the following information:


Component State

Display Name:	
Corresponding system group:	Select
Corresponding system name:	Select
Corresponding system parameter name:	
Time required for checking the display (seconds):	
Displayed component state when fail:	ON
Relational Operator:	Greater then ON
Threshold Value:	

"Display Name" - descriptive name for the control panel component state indicator. Underscore "_" should be used instead of spaces.

"Corresponding System Group" - select one of the available options.

"Corresponding System Name" - select one of the available options.

"Corresponding System Parameter Type" - select one of the available options.

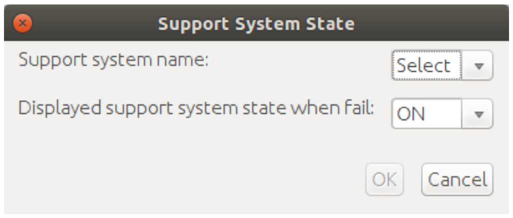
"Time Required for Checking the Display (seconds)" - used to specify the time required for the operator to read a control panel indicator.

"Displayed Value When Fail" - used to specify the value indicated by the associated indicator after a failure.

"Relational Operator" - select one of the available options.

"Threshold Value" - used to delay the actual change in the component operation state until the threshold value is exceeded.

By clicking on the "Support System State Insert" icon, a pop-up window will show up requiring the following information:


Support System State

Support system name:	Select
Displayed support system state when Fail:	ON

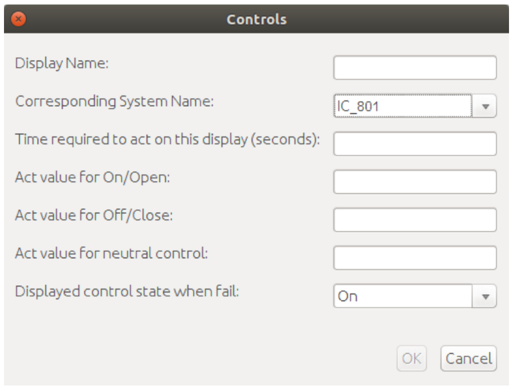
"Support System Name" - select one of the available options.

"Displayed Support System State When Fail" - used to specify the value indicated by the associated indicator after a failure.

Step 5: Create Plant Control Room Controls

In the "Plant Model -> Plant Control Room -> Controls" tab, the control panel controls are created. All the control manipulations need to be performed through the control panel interface. There are two types of controls: "simple" controls used for discrete control values (e.g. block valves) and "with fine adjust" controls used for components that can be operated over a continuous range of values (e.g. throttle valves).

By clicking on the "Controls Insert" icon, a pop-up window will show up requiring the following information:


Controls

Display Name:

Corresponding System Name: IC_801

Time required to act on this display (seconds):

Act value for On/Open:

Act value for Off/Close:

Act value for neutral control:

Displayed control state when fail: On

"Display Name" - descriptive name for the control panel control indicator. Underscore "_" should be used instead of spaces.

"Corresponding System Name" - select one of the available options.

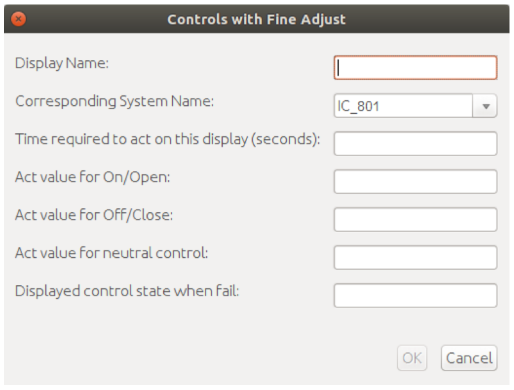
"Time Required to Act on This Display (seconds)" - used to specify the time required for the operator to manipulate this control.

"Act Value for On/Open" - used to specify the upper control limit value.

"Act Value for Off/Close" - used to specify the lower control limit value.

"Displayed Control State When Fail" - used to specify the control state indicated by the associated indicator after a failure.

By clicking on the "Controls with Fine Adjust Insert" icon, a pop-up window will show up requiring the following information:



Display Name:
Corresponding System Name: IC_801
Time required to act on this display (seconds):
Act value for On/Open:
Act value for Off/Close:
Act value for neutral control:
Displayed control state when fail:

"Display Name" - descriptive name for the control panel control indicator. Underscore "_" should be used instead of spaces.

"Corresponding System Name" - select one of the available options.

"Time Required to Act on This Display (seconds)" - used to specify the time required for the operator to manipulate this control.

"Act Value for On/Open" - used to specify the upper control limit value.

"Act Value for Off/Close" - used to specify the lower control limit value.

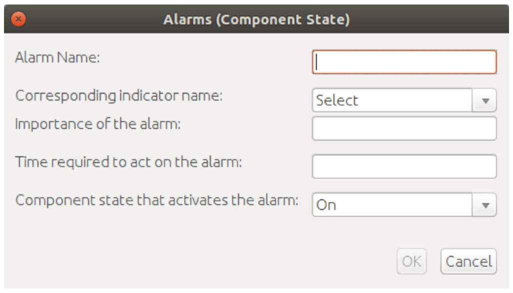
"Act Value for Neutral Control" - used to specify the neutral control value.

"Displayed Control State When Fail" - used to specify the control state indicated by the associated indicator after a failure.

Step 6: Create Plant Control Room Alarms

In the "Plant Model -> Plant Control Room -> Alarms" tab, the control panel alarms are created. All the alarm status information perceived by the crew need to be displayed on the control panel.

By clicking on the “Alarms (Component State) Insert” icon, a pop-up window will show up requiring the following information:


Alarms (Component State)

Alarm Name:

Corresponding indicator name: Select

Importance of the alarm:

Time required to act on the alarm:

Component state that activates the alarm: On

“Alarm Name” - descriptive name for the control panel alarm indicator. Underscore “_” should be used instead of spaces.

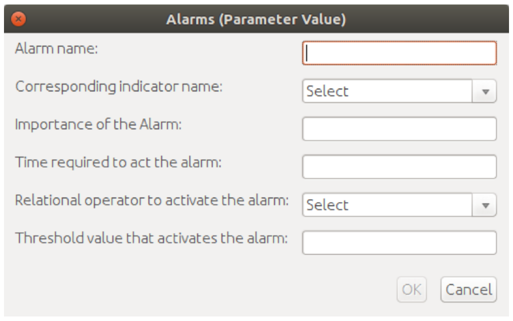
“Corresponding Indicator Name” - select one of the available options.

“Importance of the Alarm” - used to specify the weighting importance factor for the alarm.

“Time Required to Act on The Alarm” - used to specify the time required for the operator to perceive this alarm.

“Component State That Activates the Alarm” - used to specify the component state that triggers the alarm.

By clicking on the “Alarms (Parameter Value) Insert” icon, a pop-up window will show up requiring the following information:


Alarms (Parameter Value)

Alarm name:

Corresponding indicator name: Select

Importance of the Alarm:

Time required to act the alarm:

Relational operator to activate the alarm: Select

Threshold value that activates the alarm:

“Alarm Name” - descriptive name for the control panel alarm indicator. Underscore “_” should be used instead of spaces.

“Corresponding Indicator Name” - select one of the available options.

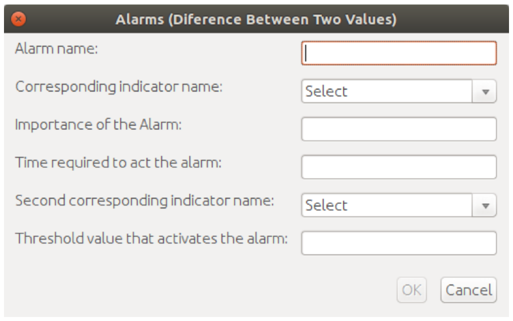
“Importance of the Alarm” - used to specify the weighting importance factor for the alarm.

"Time Required to Act on The Alarm (seconds)" - used to specify the time required for the operator to perceive this alarm.

"Relational Operator to Activate the Alarm" - select one of the available options. It is used to set up the comparison relationship that triggers the alarm.

"Threshold Value That Activates the Alarm" - used to specify the parameter value to complete the comparison relationship that triggers the alarm.

By clicking on the "Alarms (Difference Between Two Values) Insert" icon, a pop-up window will show up requiring the following information:



Alarms (Diference Between Two Values)

Alarm name:	
Corresponding indicator name:	Select
Importance of the Alarm:	
Time required to act the alarm:	
Second corresponding indicator name:	Select
Threshold value that activates the alarm:	

"Alarm Name" descriptive name for the control panel alarm indicator. Underscore "_" should be used instead of spaces.

"Corresponding Indicator Name" - select one of the available options.

"Importance of the Alarm" - used to specify the weighting importance factor for the alarm.

"Time Required to Act on The Alarm (seconds)" - used to specify the time required for the operator to perceive this alarm.

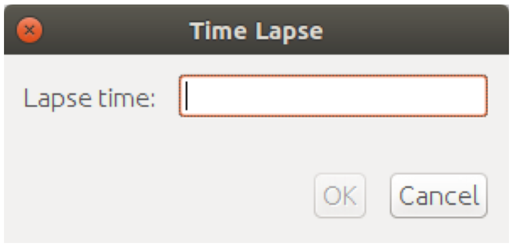
"Second Corresponding Indicator Name" - select one of the available options.

Threshold Value That Activates the Alarm: It is used to specify the one-sided deviation parameter difference alarm threshold. The alarm is activated only if the first parameter is larger than the second parameter by the threshold value.

Step 7: Create Plant Control Room SPDS

In the "Plant Model -> Plant Control Room -> SPDS Input" tab, the safety parameter display system is created. It is used to calculate the criticality of system condition PSF. This PSF represents the crew's perception of the level of degradation of the key safety functions.

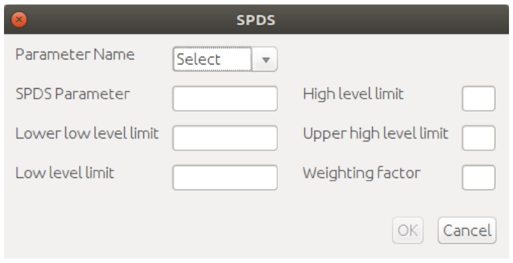
By clicking on "Time Lapse Edit" icon, a pop-up window will show up requiring an integer number. It is used to specify the time interval between safety parameter updates.



By clicking on the Insert icon, a pop-up window will show up requiring the following information:

"Parameter Name" - select one of the available options.

"Corresponding Indicator Name" - descriptive name for SPDS parameter. Underscore "_" should be used instead of spaces and the recommended name should include the prefix "SPDS_" and the associated indicator name.



"Lower Low Level Limit" - used to specify the lo_lo_level value. If the parameter value is less than the lo_lo_level, the associated PSF value is set to 10.

"Low Level Limit" - used to specify the lo_level value.

If the parameter value is between the lo_level and hi_level values, the associated PSF value is calculated from linear interpolation between the low level limits.

"High Level Limit" - used to specify the hi_level value. If the parameter value is between the lo_level and hi_level values, the associated PSF value is calculated from linear interpolation between the high level limits.

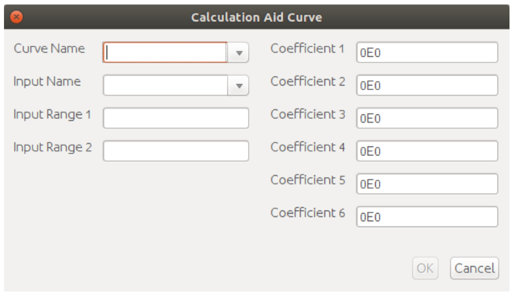
“Upper High Level Limit” - used to specify the hi_hi_level value. If the parameter value is larger than the hi_hi_level, the associated PSF value is set to 10.

“Weighting Factor” - used to specify the weighting importance factor for the SPDS.

Step 8: Create Plant Crew Calculation Aids

In the “Plant Model -> Plant Crew -> Calculation Aid Input” tab, the calculation aids are created. They are used by the crew to easily obtain a dependent variable value based on a parameter reading of an independent variable. Given the parameter reading, the calculation aid curve locates a corresponding point on the curve and provides the value of the dependent variable of this point as reference. The curves are generated with quintic polynomial functions.

By clicking on the “Insert” icon, a pop-up window will show up requiring the following information:



Curve Name
Input Name
Input Range 1
Input Range 2
Coefficient 1 0E0
Coefficient 2 0E0
Coefficient 3 0E0
Coefficient 4 0E0
Coefficient 5 0E0
Coefficient 6 0E0

“Curve Name” - descriptive name of the calculation aid curve. Underscore “_” should be used instead of spaces.

“Input Name” - select the associated indicator name.

“Input Range 1” - used to set the lower bound of the independent variable.

“Input Range 2” - used to set the upper bound of the independent variable.

“Coefficient 1 – 6” - the quintic polynomial function coefficients.

Step 9: Create Plant Crew Static PSFs

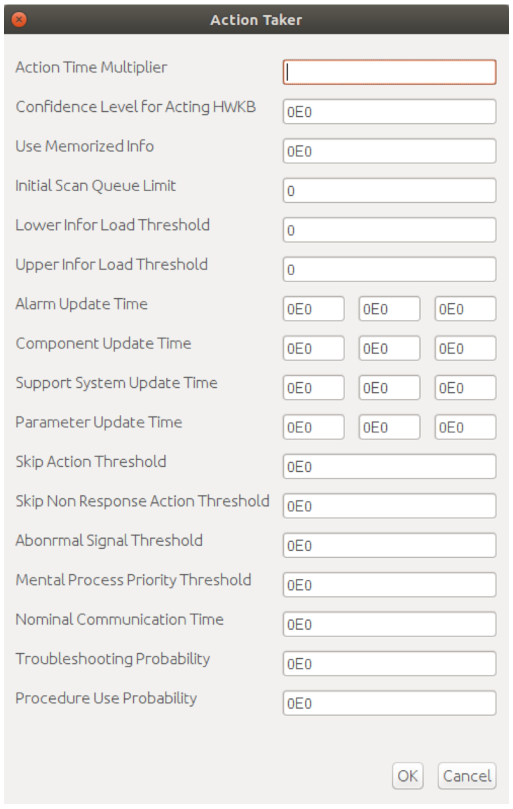
In the “Plant Model -> Plant Crew -> Static PSFs Input” tab, the crew’s static PSFs are created. They are of four types: individual related factors, team factors, organizational factors, and external factors.

By clicking on the "Action Taker," "Decision Maker" or "Consultant Insert" icon, a pop-up window will show up requiring the following information:

"Action Time Multiplier" - used to proportionally adjust the operator execution time for communication, action-taking, and decision-making activities. The action time multiplier is uniformly applied to all operator activities.

"Confidence Level for Acting HWKB" - the default confidence level for activating operator hard wired diagnoses. No dynamic event tree branches are generated by this parameter.

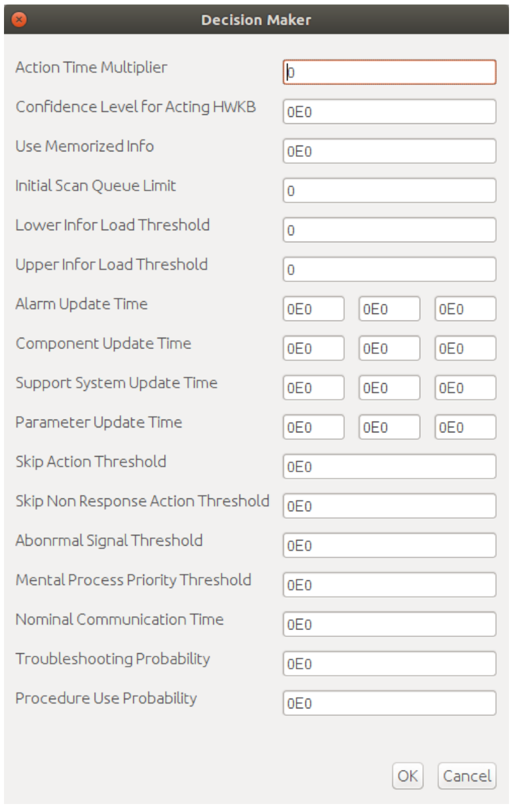
"Use Memorized Info" - used to establish the branching probability for enabling the operator's use of previously perceived and memorized plant data. This parameter establishes the branching probability for enabling the operator's use of previously perceived and memorized plant data. When the use of memorized information is enabled, the operator will use the memorized information to address data requirements of procedure expectations and knowledge-based action prerequisites. When the use of memorized information is blocked, the operator will always obtain current information from the plant control panel. The use of memorized information can reduce activity execution time but may result in the use of outdated and incorrect information. For values greater than 0.999999, the use of memorized information is always enabled. For values less than 0.000001, the use of memorized information will always be blocked. Intermediate values will cause a branching point to be generated early in the simulation where one branch enables the use of memorized information (with the branching probability set to the input value) and a second branch blocks the use of memorized information (with the branching probability set to the complement of the input value). Even when the use of memorized information is enabled, the operator may block the use of previously perceived information if it is not recent. The criteria used to judge the relevance of plant data is established in the alarm, component, and parameter update time input parameters.



Action Time Multiplier			
Confidence Level for Acting HWKB	0E0		
Use Memorized Info	0E0		
Initial Scan Queue Limit	0		
Lower Infor Load Threshold	0		
Upper Infor Load Threshold	0		
Alarm Update Time	0E0	0E0	0E0
Component Update Time	0E0	0E0	0E0
Support System Update Time	0E0	0E0	0E0
Parameter Update Time	0E0	0E0	0E0
Skip Action Threshold	0E0		
Skip Non Response Action Threshold	0E0		
Abonrmal Signal Threshold	0E0		
Mental Process Priority Threshold	0E0		
Nominal Communication Time	0E0		
Troubleshooting Probability	0E0		
Procedure Use Probability	0E0		

"Initial Scan Queue Limit" - used to set the limit of the maximum number of parameters that may be placed in the operator's scan queue. The operator periodically updates the memorized values of parameters contained in the scan queue with recent information from the control panel. A higher scan queue limit will allow the operator to monitor more parameters and obtain an improved situational assessment of the plant state. Setting a lower scan queue limit reduces the number of parameters that can be periodically monitored and allows the analyst to simulate the operator's information processing and short term memory limitations. The actual scan queue limit is dynamically adjusted during the simulation and may be less than this input value due to the influence of certain performance influencing factors. When the size of the scan queue exceeds the dynamic limit, low priority parameters are removed from the queue until the size limitation is met.

"Lower and Upper Information Load Threshold" - the lower and upper information load thresholds are used to calculate the value of the information load PSF. The information load PSF is based on the operator's average information processing rate. When the information processing rate is less than the lower threshold, the PSF value is set to 0.0. When the average information processing rate is greater than the upper threshold, the PSF value is set to 10.0. The PSF value for intermediate information processing rates is calculated from a linear interpolation between the lower and upper thresholds. A higher PSF value represents a greater operator information load. The information load threshold values can be adjusted to represent the operator's information processing capability.



Action Time Multiplier	0		
Confidence Level for Acting HWKB	0E0		
Use Memorized Info	0E0		
Initial Scan Queue Limit	0		
Lower Infor Load Threshold	0		
Upper Infor Load Threshold	0		
Alarm Update Time	0E0	0E0	0E0
Component Update Time	0E0	0E0	0E0
Support System Update Time	0E0	0E0	0E0
Parameter Update Time	0E0	0E0	0E0
Skip Action Threshold	0E0		
Skip Non Response Action Threshold	0E0		
Abonrmal Signal Threshold	0E0		
Mental Process Priority Threshold	0E0		
Nominal Communication Time	0E0		
Troubleshooting Probability	0E0		
Procedure Use Probability	0E0		

"Alarm Update Time" - these parameters establish the relevance criteria used by the operator when the use of memorized information is enabled. A three parameter Weibull distribution is used to describe the probability of old alarm state information use. When the use of memorized information is enabled, the operator will check to determine if the alarm state required to evaluate a procedural expectation of knowledge-based action prerequisite has been previously perceived. If the alarm state has been perceived, a Monte Carlo simulation is used to calculate an alarm update time. If the age of the perceived alarm state is less than the alarm update time obtained from the Monte Carlo simulation, the operator will use the memorized information. If the age of the perceived alarm state is greater than the alarm update time, the operator will obtain the current alarm state from the control panel. This parameter can be used to prevent the operator for utilizing unacceptably old information.

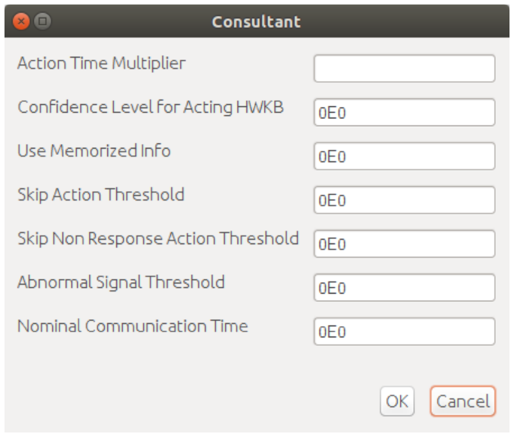
"Component Update Time" - parameters are used in a similar manner as "Alarm Update Time" parameters.

"Support System Update Time" - parameters are used in a similar manner as "Alarm Update Time" parameters.

“Parameter Update Time” - parameters are used in a similar manner as “Alarm Update Time” parameters.

“Skip Action Threshold” - the minimum probability for generating a branching point for skipping a procedural step. Procedural steps have three main parts: (1) initial action activity, (2) expectations associated with the initial action activity, and (3) a non-response action that is executed if the action expectations are not met. The operator may skip either the initial action activity or the non-response action (evaluation of the action expectations cannot be skipped). The probability of skipping the initial action or non-response action is dynamically calculated based upon the baseline skip probability for the step component (specified in the procedure step input file), the type of procedure being followed, the step objectives, the relevance of the action to the operator’s situational assessment, and certain PSFs. If the calculated skip probability for the initial action activity exceeds the skip action threshold, a branching point with two branches is generated. The procedure step initial action activity and associated expectation evaluation are skipped for skipped step branching path, while the step is executed on the complimentary branching path. The branching probability for the step skipping branch is set equal to the calculated branch probability and the branch probability for execution of the step is equal to the complement of the skip probability. If the calculated skip probability is less than the skip action threshold, the associated action is executed and no branching point is generated. The analyst can reduce the excessive generation of procedure step skipping branches by setting the skip action threshold to a higher value. Setting this parameter equal to 1.0 will prevent the operator from skipping procedural actions (i.e., all procedure initial action activities are executed).

“Skip Non Response Action Threshold” - the minimum probability for generating a branching point for skipping a non-response action in a procedure step.


Consultant

Action Time Multiplier	
Confidence Level For Acting HWKB	0E0
Use Memorized Info	0E0
Skip Action Threshold	0E0
Skip Non Response Action Threshold	0E0
Abnormal Signal Threshold	0E0
Nominal Communication Time	0E0

"Abnormal Signal Threshold" - used to establish the diagnostic threshold for an abnormal condition. The ADS-IDAC model includes a fuzzy logic diagnostic engine that supports the operator's situational assessment of the plant state. This parameter effectively establishes the operator's sensitivity to detecting an abnormal event. If the parameter is set to a high value, the operator will require more information to support an abnormal condition diagnosis. A lower value reduces the information requirements for detecting an abnormal condition, but might result in the operator reaching a "false positive" conclusion for an abnormal event. Following the diagnosis of an abnormal event, the operator will suspend the execution of all low priority mental procedures. The diagnosis of an abnormal condition also influences the goal selection process.

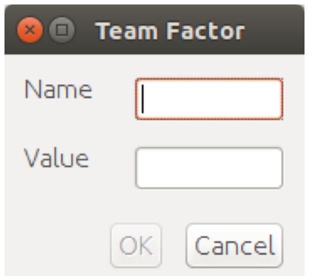
"Mental Process Priority Threshold" – threshold for operator suspending all low priority mental procedures following the identification of an abnormal condition (based on the abnormal signal threshold and diagnostic engine) or a reactor trip. High priority procedures are associated with a low priority value (i.e., the highest priority procedures have a priority value of "1"). Once an abnormal condition has been detected, the operator suspends the execution of all mental procedures will a lower priority level than the specified threshold value (i.e., procedures with a priority value higher than the threshold). The purpose of this parameter is to allow the operator to interrupt mental procedures that are no longer appropriate following a reactor trip or during an accident event. Because the abnormal condition diagnosis is not reset, the suspension of low priority mental procedures can occur only once during an accident sequence. Thus, low priority mental procedures that are activated following the initial diagnosis of a reactor trip or abnormal event are not automatically suspended and may be executed.

"Nominal Communication Time" - time required to perform inter-crew communication. Certain crew activities require coordination and communication between the control room operators. For example, only the Decision Maker can direct the performance of proceduralized actions and only the Action Taker can manipulate the control panel. Therefore, the execution of a procedure step requires the "Decision Maker" to direct the "Action Taker" to perform the specified action followed by a report from the "Action Taker" to the "Decision Maker" that the action had been accomplished. The parameter establishes the communication delay time (in seconds) and is controlled by the sender of the information.

"Troubleshooting Branch Probability" - branching probability for activating the troubleshooting goal. If the value is greater than 0.999999, the operator will always activate the troubleshooting goal if the normal operation goal is no longer appropriate. If the value is less than 0.000001, the operator will bypass the troubleshooting goal and always activate the monitoring goal when the normal operation goal is no longer appropriate. For intermediate values, a branching point will be generated with two operator goal branches. One branch will activate the troubleshooting goal with a branch probability equal to the troubleshooting probability. The other branch will activate the monitoring goal with a complimentary branching probability. When the troubleshooting goal is activated, the crew can implement knowledge-based actions to address the abnormal condition. This parameter is currently implemented only for the "Decision Maker." A dummy value should be entered for the "Action Taker."

"Procedure Use Probability" - branching probability for enabling a transition from the troubleshooting goal to the "maintain global safety" goal upon the identification of a reactor trip condition. If this value is greater than 0.999999, the operator will always transition from the troubleshooting goal to the "maintain global safety" goal following the identification of a reactor trip condition. If this value is less than 0.000001, the goal transition from troubleshooting to "maintain global safety" margin is blocked. For intermediate values, a branching point will be generated with two operator goal branches. One branch will activate the transition from the troubleshooting goal to the "maintain global safety margin" goal with a branch probability equal to the troubleshooting probability. The other branch will block the transition to the "maintain global safety margin" goal. Effectively, this parameter allows the crew to transition from a knowledge-based approach to accident mitigation to a procedure following approach. This parameter is currently implemented only for the "Decision Maker." A dummy value should be entered for the "Action Taker."

By clicking on the "Team Factor", "Organizational Factor" or "External Factor Insert" icon, a pop-up window will show up requiring the following information:



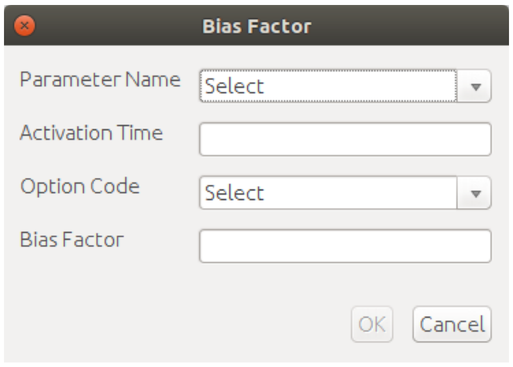
"Name" - descriptive name for the PSF.

"Value" - : Value of the PSF.

Step 10: Create Plant Crew Bias Factors & Safety Parameters

In the "Plant Model -> Plant Crew -> Action Taker, Decision Maker or Consultant -> Bias Factors & Safety Parameters Input" tab, the bias factors and the safety parameters are created. The bias factors are used to allow the analyst to simulate failed control panel instrumentation. In general, when an operator perceives the value of a plant indicator, the actual value of the parameter is obtained directly from the RELAP thermal-hydraulic model. In order to simulate a failed indicator, it is necessary to apply a bias factor to the RELAP generated data in order to model an instrument failure. Several indicator failure options are available, including additive errors, proportional errors, stuck instrumentation, and instruments that cannot read above or below a set threshold. When a parameter is biased in this manner, the operator may use inaccurate data when assessing the procedure step expectations, knowledge-based action prerequisites, and activation criteria for hard wired mental beliefs.

By clicking on the “Bias Factor Insert” icon, a pop-up window will show up requiring the following information:



“Parameter Name” - select one of the available options.

“Activation Time” - activation time for parameter bias factor. If the simulation time is less than the activation time, the parameter is unbiased and the operator will perceive the actual parameter value when the component is read from the control panel. If the simulation time is greater than the activation time, the parameter will be biased by the specified bias factor.

“Option Code” - select one of the available options.

“Bias Factor” - bias factor that will be applied to the perceived parameter based on the bias option code specified by the analyst. The analyst should ensure that the bias factor value and units are consistent with the normal output range for the associated indicator.

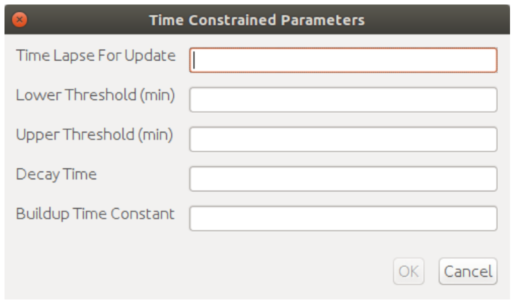
For “Safety Parameters” see ***Step 7: Create Plant Control Room SPDS.***

Step 11: Create Plant Crew Time Constrained and Scanned Parameters

In the "Plant Model -> Plant Crew -> Action Taker, Decision Maker or Consultant -> Time Constrained & Scanned Parameters Input" tab, the time constrained and scanned parameters are created.

Each operator profile includes data to define how the time constraint load PSF value is calculated. The profile contains a listing of plant parameters used to calculate the time constraint PSF value along with the associated critical threshold values. Two different threshold levels are used to calculate the PSF value: a normal operation and an accident threshold. When the operator's high level goal is maintaining normal operation or troubleshooting an abnormal condition, the normal operation threshold is used. If the operator switches to the goal of mitigating an accident condition, the time constraint PSF value is based on the accident threshold. The use of two different threshold values allows ADS-IDAC to capture an operator's changing sensitivity to key parameters depending on the overall perceived plant condition. In general, the normal accident threshold is set to a level corresponding toNPP trip set points. The accident level threshold is normally set to a less restrictive value that is more indicative of the availability of a key safety function.

By clicking on "Time Constrained Parameters Edit" icon, a pop-up window will show up requiring the following information:



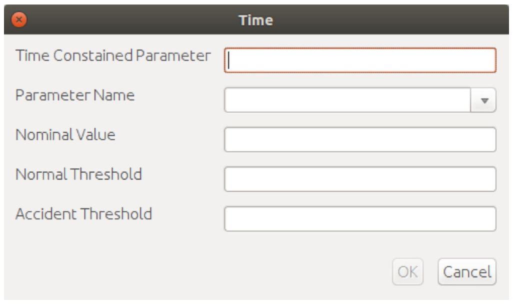
"Time Lapse For Update" - time increment (in seconds) between successive updates in time constrained parameter values. Because time constrained parameters are based on perceived information, the update uses the operator's currently memorized value. If the operator has not perceived a new parameter value since the last update, an updated value is calculated from the perceived rate of change of the parameter value.

“Lower Threshold and Upper Threshold (min)” - used to calculate the time constraint PSF for each monitored parameter. If the time to reach the applicable threshold value is less than the lower threshold, the parameter PSF value is set to 10.0, and if it is larger than the upper threshold, the parameter PSF value is set to 0.0. If the time to reach is between the lower threshold and upper threshold, the PSF value is interpolated based on the parameter value and rate of change.

“Decay Time” - specifies the decay constant to be applied to the parameter PSF value when the associated parameter exceeds the accident threshold.

“Buildup Time Constant” - specifies the buildup time constant to be applied to the parameter PSF value. This parameter allows a more realistic buildup and decay of time constrained induced stress.

By clicking on the “Time Constrained Insert” icon, a pop-up window will show up requiring the following information:



“Time Constrained Parameter” - descriptive name for the time constrained parameter. Underscore “_” should be used instead of spaces and the recommended name should include the prefix “TCL_” and the associated indicator name.

“Parameter Name” - select one of the available options.

“Nominal Value” - used to specify the nominal value for the monitored parameter. This value is used to determine if the normal parameter value is above or below the applicable threshold values.

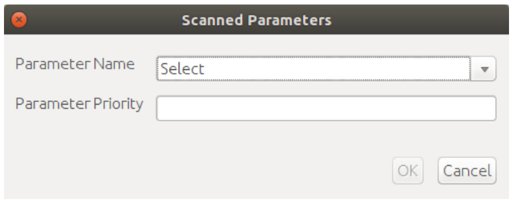
“Normal Threshold” - used to specify the parameter threshold value used during normal operation.

“Accident Threshold” - used to specify the parameter threshold value used during emergency operation.

The ADS-IDAC quantitative perception filter models the information scanning process used by operators to monitor plant status. In general, the control room operators frequently monitor a small subset of plant instrumentation. When plant conditions degrade, the operators may add additional parameters to their scanning in response to procedural requirements or alarms. The more parameters the operator can monitor, the more accurate their status assessment is. However, the operator does not have an infinite capacity to monitor plant instrumentation. Consequently, there are limits to how many items and operator can effectively monitor. External and internal factors (e.g., stress or time pressure) may force the operator to reduce the number of parameters monitored. Consequently, operators may limit monitoring activities to a focused set of items that are most pertinent to the perceived NPP state.

Within the ADS-IDAC model, the focusing process is controlled by the operator’s control panel “scan queue.” The scan queue contains a listing of parameters that the operator monitors on a frequent basis. Scan queue parameters include parameters, alarms, component and support system states. The number of items contained in the scan queue is limited by the individual capabilities of the operator, the amount of attention the operator can apply to information gathering, and the operator’s perception of the current plant state. As the number of monitored items in the scan queue increases, the operator improves their ability to accurately assess and diagnosis the plant state.

By clicking on the “Scanned Parameters,” “Scanned Components,” “Scanned Alarms,” and “Scanned Support Systems Insert” icon, a pop-up window will show up requiring the following information:



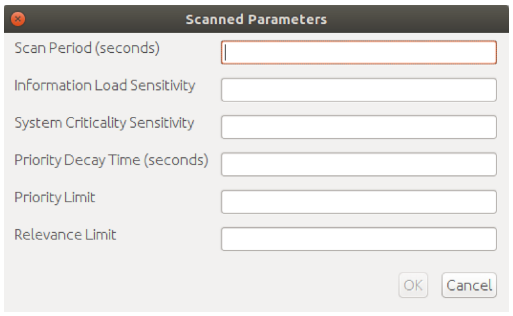
"Parameter Name" - select one of the available options.

"Parameter Priority" - used to specify the parameter's initial priority level. A lower value designates a higher priority (i.e., the highest priority items are designated with a priority level of 1). Because the operator's priority level can decay over time, specifying a low value will increase the residence time of the parameter on the scan queue list.

By clicking on the "Scanned List Parameters Edit" icon, a pop-up window will show up requiring the following information:

"Scan Period (seconds)" - used to specify the time delay between control panel monitoring scans. A lower value will improve the operator's plant state assessment making parameter updates more frequent. More frequent control panel scanning increases the operator's information load and may limit the number of parameters that can be monitored. Longer scan periods will minimize the information load but may result in the operator relying on outdated information - particularly during rapidly evolving plant events.

"Information Load Sensitivity" - used to specify the sensitivity of the scan queue size limit to the operator's information load. Because the maximum information load PSF value is 10.0, the information load sensitivity factor is limited to values no greater than 0.01.


Scanned Parameters
Scan Period (seconds)
Information Load Sensitivity
System Criticality Sensitivity
Priority Decay Time (seconds)
Priority Limit
Relevance Limit
OK
Cancel

"System Criticality Sensitivity" - used to specify the sensitivity of the scan queue size limit to the operator's perceived criticality of the plant state. Because the maximum criticality of system condition PSF value is 10.0, this factor is limited to values no greater than 0.01.

"Priority Decay Time (seconds)" - used to specify the length of the delay interval for successive reductions in scanned item priority.

"Priority Limit" - used to specify the lowest item priority level that can be maintained in the operator's scan queue.

"Relevance Limit" - used to specify the threshold value used by the operator model to determine if an imbalance event diagnosis is relevant. If the maximum imbalance event diagnostic score for the functions supported by the item is greater than this value, the scanned item will be considered relevant to the operator. If the scanned item is not relevant to the operator, the item's priority level will be decreased one increment during every priority decay time interval.

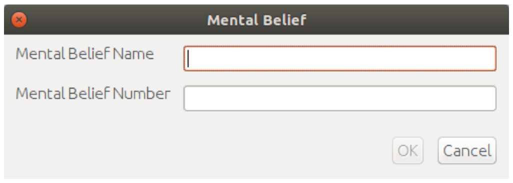
Step 12: Create Plant Crew Hardwired Diagnoses

In the "Plant Model -> Plant Crew -> Action Taker, Decision Maker or Consultant -> Hardwired Diagnosis Input" tab, the hardwired mental beliefs that may be activated by the operator during the simulation are created.

Once a mental belief is activated, the operator may use it to initiate a mental procedure, activate other mental beliefs, or help evaluate procedure step expectations and knowledge-based action prerequisites. Mental beliefs are activated based on satisfying the specified prerequisite alarm states, component states, parameter states, support system states, control value states, mental belief states, and procedure activations. The confidence level for a mental belief calculated from the ratio of satisfied prerequisite states to the total number of prerequisites specified for the mental belief.

A mental belief is activated when the following conditions have been met: (1) the mental belief confidence is greater than the specified activation confidence, (2) mental belief activation is not blocked by the reset time delay, and (3) the mental belief has been enabled by specifying a branching probability greater than 0.000001.

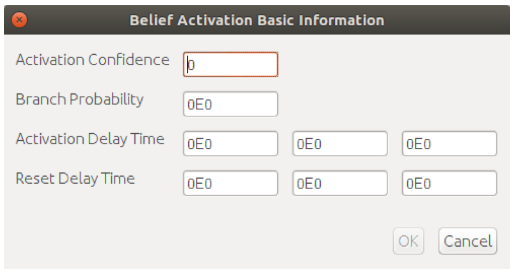
By clicking on the "Mental Belief Insert" icon, a pop-up window will show up requiring the following information:



"Mental Belief Name: - descriptive name for the mental belief. Underscore "_" should be used instead of spaces. If the mental beliefs "Normal_Operation" and "Reactor_Tripped" exist, they are also utilized for the goal selection process.

"Mental Belief Number" - mental belief indexing number.

By clicking on the "Belief Activation Basic Information Edit" icon, a pop-up window will show up requiring the following information:


Belief Activation Basic Information

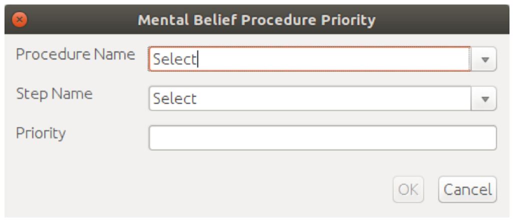
Activation Confidence	0		
Branch Probability	0E0		
Activation Delay Time	0E0	0E0	0E0
Reset Delay Time	0E0	0E0	0E0

"Activation Confidence" - minimum confidence level necessary to enable mental belief activation and branch generation. The mental belief confidence is equal to the ratio of satisfied conditions to the total number of prerequisite conditions. The activation confidence level influences the activation logic for the associated mental belief.

"Branch Probability" - branching probability for activating the mental belief. If the value is greater than 0.999999, a single branch will be generated to activate the mental belief provided the minimum confidence level is met and the reset time delay has not blocked activation. If the value is less than 0.000001, the operator will bypass mental belief activation and the associated mental procedure (if supplied) will not be initiated. For intermediate values, a branching point will be generated with two mental belief branches. One branch will activate the mental belief (and initiate the associated mental procedure if applicable) with a branch probability equal to the branch probability. The other branch will bypass mental belief activation with a complimentary branching probability. A bypassed mental belief may become reactivated once the reset time delay has elapsed (provided the appropriate prerequisite conditions are met).

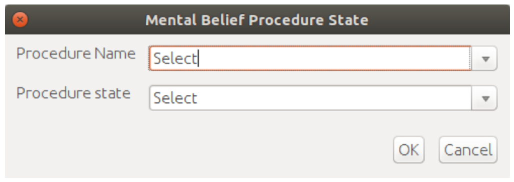
"Activation Delay Time' - parameters that establish the activation delay time for initiation of the associated mental procedure (if supplied). Upon activation of a mental belief, the associated mental procedure will be added to the operator's procedure queue. However, a mental procedure will not be initiated until the activation delay time has elapsed. This parameter allows the analyst to separate (in time) the activation of the mental belief and the execution of the associated mental procedure. A three parameter Weibull distribution is used to describe the probability of the use of activation time delay. Upon activation of a mental belief, a Monte Carlo simulation is used to calculate the mental procedure activation time delay. Once the activation time delay has elapsed, the mental procedure can be initiated by the operator.

"Reset Delay Time" - parameters that establish the reset delay time for a mental belief. The purpose of the reset delay time is to allow the analyst to create mental beliefs that can be activated multiple times during a simulation run. The reset delay time provides a dormancy time during which the mental belief cannot be reactivated, even if the prerequisite conditions are met.



This provides more realistic control over certain repetitive operator actions such as control of auxiliary feed water flow rate. It is modeled similarly to the activation delay time.

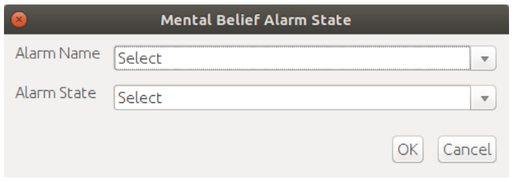
By clicking on the "Mental Procedure Priority Edit" icon, a pop-up window will show up requiring the following information:



"Procedure Name" - select one of the available options.

"Step Name" - select one of the available options.

"Priority" - used to specify the priority level of the associated mental procedure. If no mental procedure is associated with the mental belief, a dummy integer value should be entered. Lower priority numbers indicate a higher procedure priority.



By clicking on the "Procedures States Insert" icon, a pop-up window will show up requiring the following information:

"Procedure Name" - select one of the available options.

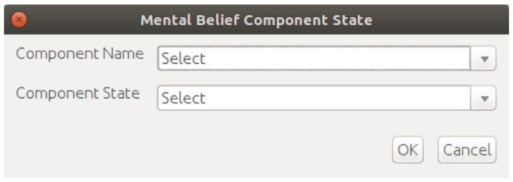
"Procedure State" - select one of the available options.

By clicking on the "Alarms States Insert" icon, a pop-up window will show up requiring the following information:

"Alarm Name" - select one of the available options.

"Alarm State" - select one of the available options.

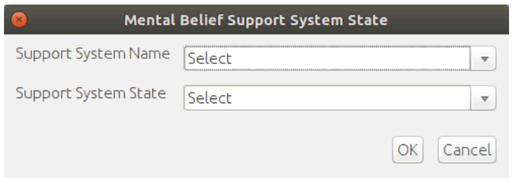
By clicking on the "Components States Insert" icon, a pop-up window will show up requiring the following information:



"Component Name" - select one of the available options.

"Component State" - select one of the available options.

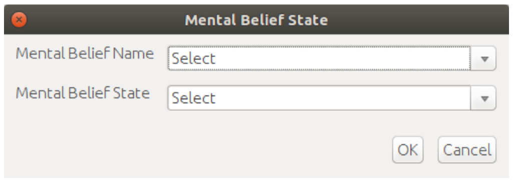
By clicking on the "Support Systems States Insert" icon, a pop-up window will show up requiring the following information:



"Support System Name" - select one of the available options.

"Support System State" - select one of the available options.

By clicking on the "Mental Beliefs States Insert" icon, a pop-up window will show up requiring the following information:



"Mental Belief Name" - select one of the available options.

"Mental Belief State" - select one of the available options.

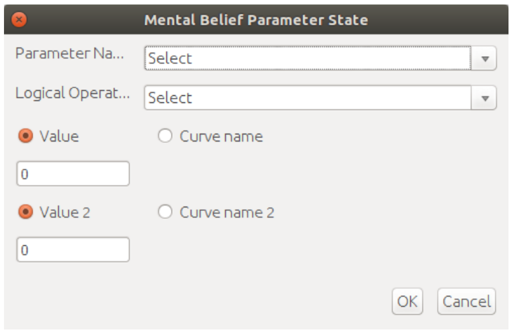
By clicking on the "Parameters States Insert" icon, a pop-up window will show up requiring the following information:

“Parameter Name” - select one of the available options.

“Logical Operation” - select one of the available options.

Select and insert Value or Curve Name.

By clicking on the “Control Value Insert” icon, a pop-up window will show up requiring the following information:

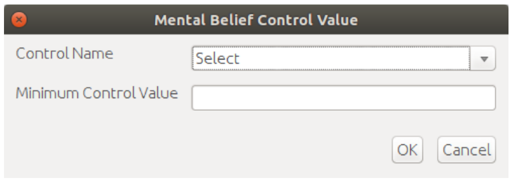


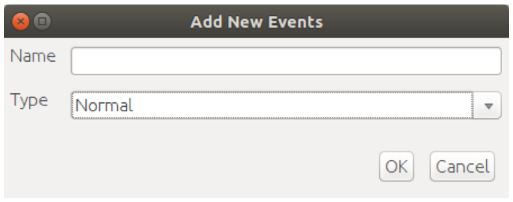
“Control Name” - select one of the available options.

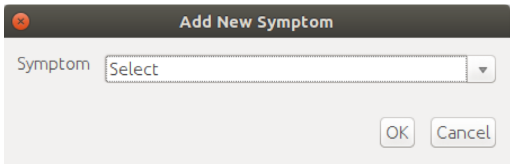
“Minimum Control Value” - is used to specify the threshold control value necessary to satisfy the prerequisite condition.

Step 13: Create Plant Crew Event Matrix

In the “Plant Model -> Plant Crew -> Action Taker, Decision Maker or Consultant -> Event Matrix Input” tab, a listing of symptoms used to support event diagnosis and a relationship value matrix are created. The relationship value matrix provides the linkage between symptoms and events that are used to generated diagnostic results. Four general types of event types are considered: normal operating events (“normal”), anticipated operational occurrences (“AOO”), design basis accidents (“DBA”), and mass, energy, or momentum flow imbalances (“Imbalance”). Because the diagnostic engine uses a fuzzy logic inference approach,







the relationship values that link symptoms and events should be viewed as set membership values rather than a strict confidence level or subjective probability.

By clicking on the “Symptom Insert” icon, a pop-up window will show up requiring the following information:

“Symptom” - select one of the available options.

By clicking on the “Event Matrix Insert” icon, a pop-up window will show up requiring the following information:

“Name” - descriptive name for the event. Underscore “_” should be used instead of spaces.

“Type” - select one of the available options.

Step 14: Create Plant Crew System Decomposition

In the “Plant Model -> Plant Crew -> Action Taker, Decision Maker or Consultant -> System Decomposition Input” tab, the functional system decomposition is created.

To model an operator’s mental model of the reactor plant, a functional decomposition of reactor plant systems and components is used to link parameters, components, alarms, and controls that support similar functions. The system decomposition supports two main features in the ADS-IDAC model: (1) skipping of procedure steps, and (2) management of the operator scan queue. These features are supported by the assignment of a “relevance factor” (from 0.0 – 1.0) to each component based on the operator’s perceived assessment of the plant state. Items that are considered relevant to the current plant state assessment are assigned a high relevance factor, while the remainder are assigned a low value. The relevance is based on the diagnosis score for the functional imbalance(s) associated with the component. If the operator has diagnosed that a specific imbalance event has occurred, items associated with the functional imbalance will be assigned high relevance score. Actions associated with items relevant to the plant state assessment will be less likely to be skipped, and vice versa. Similarly, the priority level of items in the scan queue list that are relevant will remain at a higher level and are more likely to be retained in the scan queue.

ADS-IDAC utilizes a functional component categorization based on the flow of energy, mass, and momentum. In this modeling scheme, the NPP is viewed as a collection of mass, energy, and momentum flow paths - each containing sources and sinks. In general, the following rules are used to identify mass, energy, and momentum imbalances:

Energy flow imbalances are generally indicated by changes in temperature for subcooled single phase systems and changes in pressure for saturated two phase systems.

Imbalances between mass sources and sinks are generally related to net inventory measures such as tank or vessel levels.

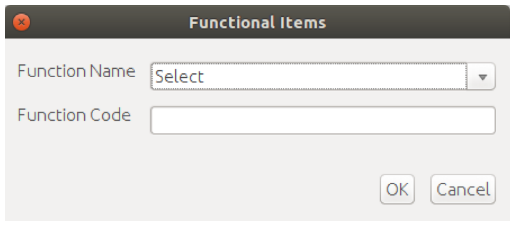
Momentum imbalances are generally indicated by changes in flow rates.

This modeling technique provides a powerful mechanism for linking components within a functional framework. To functionally categorize plant components, it is first necessary to identify the flow path boundaries. Plant system groups are used to represent the boundaries for mass, energy, and momentum flow paths. In general, it is desirable to make the plant system group boundaries as broadly as possible to maximize the ability to link plant components within the operator knowledge base.

Strong coupling between NPP systems presents a significant challenge when identifying functional system groups. Energy flow is often carried by moving fluids such as the reactor coolant or main steam systems. Changes in mass flow rate can directly impact energy flow. Consequently, coupling can result in imbalances in one flow type influencing a second flow type within the same system group or a connected system group. Coupling can also mask the cause of disruption in energy, mass, or momentum flow. For example, changes in reactor coolant system temperature due to an imbalance between reactor core power and turbine load (an energy flow imbalance) can result in variations in system volume due to the expansion or contraction of the coolant (which might be interpreted as a mass flow imbalance). An additional consideration is the diagnostic capability afforded by the system groupings. It is desirable to constrain the system group boundaries such that a flow imbalance within a grouping can be linked to a manageable number of potential causes. In practice, the identification of the system groups requires a balance between maximizing the linkage between plant components, minimizing undesirable coupling, and providing a high level of diagnosticity.

Each operator knowledge base includes a unique component functional map to match operator behavior with to a desired level of knowledge, skills, and abilities. A three-parameter coding scheme is used to identify component functions. The first parameter identifies the type of flow (i.e., energy, mass, or momentum). The second parameter identifies the system group that transports the energy, mass, or momentum flow. The third parameter identifies how the component affects (or is associated with) the flow balance in the system group.

By clicking on the “Functional Items Insert” icon, a pop-up window will show up requiring the following information:


Functional Items

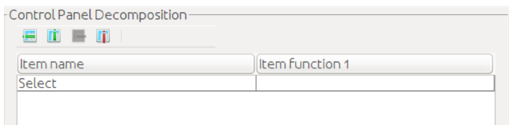
Function Name: Select

Function Code:

“Function Name” - select one of the available options.

“Function Code” - used to specify a unique identifying code for each function. Each code should be unique (codes should not be associated with more than one function). It is recommended that the functional codes be assigned using a consistent framework such as using a multi-digit integers where each digit position has a defined purpose (e.g., first digit refers to function, second digit refers to plant system, third digit refers to imbalance trend, etc.).

By clicking on the "Control Panel Decomposition Insert" icon, a pop-up window will show up requiring the following information:


Control Panel Decomposition

Item name	Item function 1
Select	

"Item Name" - select one of the available options.

"Item Function" - more than one function code may be entered for a component. It is not necessary to enter a function code, but if no codes are specified, the associated control panel item will always have a relevance score of 0.0. If an item is assigned the integer code "900" for the "Not_Applicable" function (a special text code recognized by ADS-IDAC), the relevance factor for the item will be set equal to 1.0. Each item data entry line should be terminated with the integer code "999". This code is recognized by ADS-IDAC as the termination of the item functional description. A "999" termination entry should be used even if no function codes are entered for the item.

Step 15: Create Plant Crew Diagnosis Actions

In the "Plant Model -> Plant Crew -> Decision Maker -> Diagnosis Actions Input" tab, diagnosis actions are created. Diagnosis actions are used to implement the heuristic knowledge-based problem solving approach when the control room crew is implementing the "troubleshooting" goal. This problem-solving approach is intended to model heuristic knowledge-based actions that an operator might perform outside the scope of the EOPes. Diagnostic actions are grouped within functional areas that are aligned with the imbalance events described in the Event Matrix. Because many possible actions may be available to address a specific imbalance event, each diagnosis action is assigned a priority level and a set of prerequisite conditions. Based on their perception of the plant state, operators might execute actions they believe to be reasonable given their situational assessment, but are not necessarily covered by plant procedures. Examples of such actions include reducing reactor coolant system water injection when pressurizer level is high or decreasing the steam dump rate when steam generator pressure is low. Within ADS-IDAC, heuristic knowledge-based actions can be activated based on the operator's perceived plant state, when the event membership value of a functional imbalance diagnosis exceeds a pre-defined threshold value. For example, an imbalance diagnosis of "low mass in the reactor coolant system" might lead an operator to increase reactor coolant system injection flow, reduce normal letdown flow, or actuate emergency core cooling systems. Heuristic knowledge-based actions have the following characteristics and properties in the ADS-IDAC model:

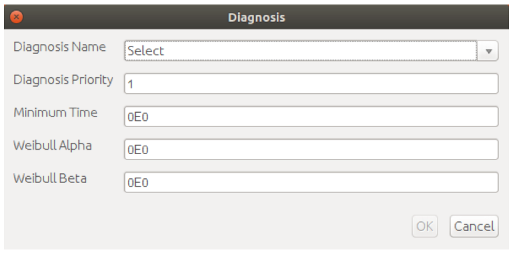
Action rules are organized within functional imbalance diagnostic groups. Each possible functional imbalance event can be associated with a list of actions intended to mitigate the associated mass, energy, or momentum imbalance.

Each functional imbalance diagnosis group is assigned a priority level to reflect the relative importance of the associated actions to the operator. For example, actions intended to address inadequate core cooling might be sequenced before actions to address low steam generator inventory in a single steam generator. The priority can be adjusted to reflect an operator's knowledge, experience, and problem solving style.

Each action can be assigned a set of prerequisite conditions that must be met prior to execution of the action. Prerequisites are used to better model the heuristic rules an operator might use to activate a specific action. Prerequisites can be associated with plant parameters, component states, alarms, active procedures in use, or an operator's mental beliefs.

Once an action in a functional diagnosis group has been activated, further actions within the functional area will be blocked for a pre-defined dormancy period. The dormancy period allows the operator to address other, possibly lower priority, functional areas.

By clicking on the “Diagnosis Insert” icon, a pop-up window will show up requiring the following information:



Diagnosis

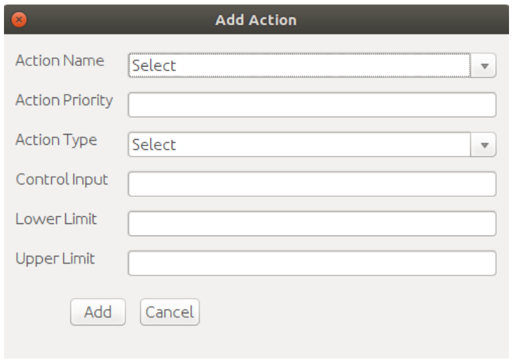
Diagnosis Name	Select
Diagnosis Priority	1
Minimum Time	0E0
Weibull Alpha	0E0
Weibull Beta	0E0

“Diagnosis Name” - select one of the available options.

“Diagnosis Priority” - the priority level of the associated imbalance diagnosis name. A lower value designates a higher priority imbalance diagnosis (i.e., the highest priority items are designated with a priority level of 1).

“Minimum Time, Weibull Alpha and Beta” - specify the: (1) dormancy time for the associated diagnosis actions following activation, and (2) the required time to evaluate action prerequisites. To capture the uncertainty associated with these times, a three parameter Weibull probability distribution is used.

By clicking on the “Actions Insert” icon, a pop-up window will show up requiring the following information:



Add Action

Action Name	Select
Action Priority	
Action Type	Select
Control Input	
Lower Limit	
Upper Limit	

“Action Name” - select one of the available options.

“Action Priority” - the priority level of the associated action name. A lower value designates a higher priority action (i.e., the highest priority items are designated with a priority level of 1).

“Action Type” - select one of the available options.

"Control Input" - value should be consistent with the control range of the controller specified by the action name variable.

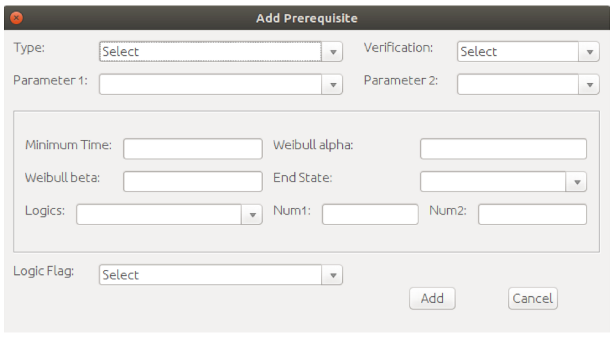
"Lower Limit" - lower control limit for the action name controller. Actions that would cause the controller position to decrease below the lower control limit are not performed.

"Upper Limit" - upper control limit for the action name controller. Actions that would cause the controller position to increase above the upper control limit are not performed.

By clicking on the "Prerequisites Insert" icon, a pop-up window will show up requiring the following information:

"Type" - select one of the available options.

"Verification" - select one of the available options. When the operator is relying on memorized information, prerequisites may be evaluated using old information - particularly during dynamic situations. Although this may model real operator behavior in certain situations, there are times when operators would be expected to re-verify control panel indicators even when the indicator had been perceived earlier. To force the operator to perform this reverification, the verification variable can be set to "YES". If the verification variable is set to "NO" or "NONE," the operator will not re-verify the control panel indications and will always use previously perceived information (if available). If the operator has not previously perceived the indicator value or state (or if the information is deemed to be too old), the operator will re-verify the information regardless of the value of the verification variable.



“Parameter 1” - select one of the available options.

“Parameter 2” - select one of the available options.

“Expected State” - select one of the available options.

“Relationship” - select one of the available options.

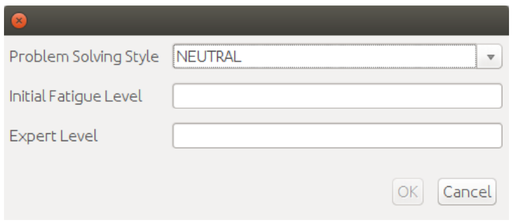
“Num1” and “Num2” - should be consistent with the range associated with the type.

“Logics” - select one of the available options. The logic flag is used to establish the Boolean relationship between successive prerequisite conditions. If the logic flag is set to “OR”, the current prerequisite condition and the next prerequisite will be connected with an “OR” gate logic to form one prerequisite unit. When the logic flag is set to “AND”, the current prerequisite and the next prerequisite are treated as separate prerequisite units and are connected by “AND” gate logic. There is no limit on the number of individual prerequisites that can be connected with an “OR” and “AND” gate logic. Thus, it is possible to create complex prerequisite requirements.

Step 16: Create Plant Crew Reasoning Machine Profile

In the “Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Profile Input” tab, the operator’s profile for knowledge-based reasoning is created.

By clicking on the “Profile Edit” icon, a pop-up window will show up requiring the following information:



Problem Solving Style	NEUTRAL
Initial Fatigue Level	
Expert Level	

"Problem Solving Style" - select one of the available options. Different problem-solving styles are currently supported. "Vagabond" tends to jump from issue to issue without satisfactory resolution of any. "Hamlet" tends to consider many possible explanations of observed findings. "Garden Path" shows excessive persistence on a single issue or activity. "Neutral" tends to have a neutral investigation style.

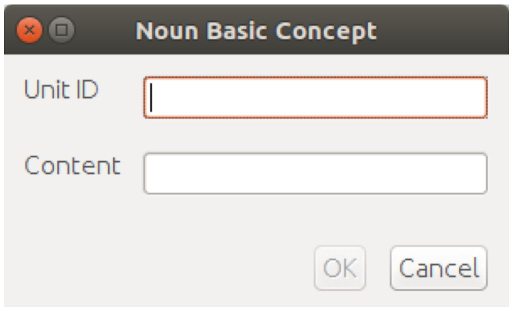
"Initial Fatigue Level" - the initial fatigue level used to initialize the fatigue PSF.

"Expert Level" - operator's general expertise level. This influences the task load, task complexity and working memory span PSFs.

Step 17: Create Plant Crew Reasoning Machine Concept Base

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Concept Base Input" tab, the operator's concept base for knowledge-based reasoning is created.

In the reasoning machine, the operator's memory information is mainly represented in a semantic way. The concept base stores the semantic elements, which are used to construct higher level semantic representations (i.e. sentences). There are two types of elements stored: basic concept unit and composed concept unit. A composed concept unit is composed of more than one basic concept units.



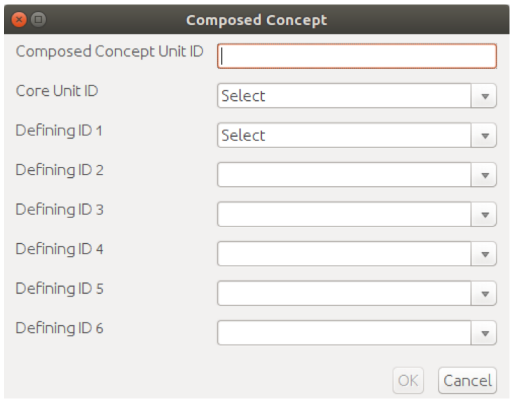
By clicking on the "Noun Basic Concepts Insert" icon, a pop-up window will show up requiring the following information:

"Unit ID" - ID of the basic concept unit. Each ID must be unique.

"Content" - the meaning of this basic concept unit.

Similarly, "Process," "Comparison Relationship," "Component State," and "Attribute Basic Concepts" can be created.

By clicking on the "Composed Concepts Insert" icon, a pop-up window will show up requiring the following information:



"Composed Concept Unit ID" - the ID of the composed concept unit. Each ID must be unique. Underscore "_" should be used instead of spaces.

"Core Unit ID" - select one of the available options. Each composed concept unit is composed of several other member concept units. One composed concept unit has one core concept unit, and the others are used to define the composed concept unit. Each member concept unit of a composed concept unit must be defined in earlier position before this composed concept unit.

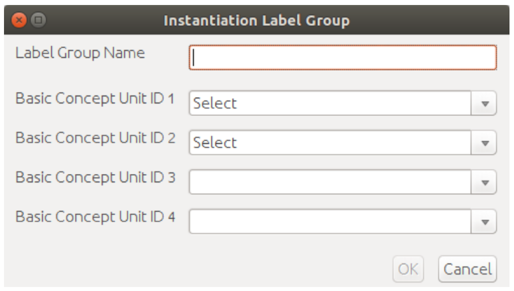
"Defining ID" - select one of the available options.

Step 18: Create Plant Crew Reasoning Machine Instantiation Base

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Instantiation Base Input" tab, the operator's instantiation base for knowledge-based reasoning is created.

It allows the user to use some indefinite concept units to build general knowledge links, which could be automatically applied to its instance concept units. For example, "The feed water flow in X loop is smaller than the main steam flow in X loop" causes "The water level in steam generator X decreases". In this input file, there are two types of information. One is called "label group" and the other is called “instance group”. The label group lists a group of label concept units, which could be used to substitute the indefinite concept unit “X” and to compose more complex concept units. An instance group has one indefinite composed concept unit and one or more stances of this composed concept unit. Each instance group links to a label group to indicate what is varying among its instances. An example instance group (SG_X, SG_A, SG_B, SG_C) links to a label group name “loop” with members (A, B, C). Any knowledge link built with SG_X could be applied to SG_A, SG_B, and SG_C. The difference among SG_A, SG_B and SG_C is that they belong to different loops.

By clicking on the “Instantiation Label Groups Insert” icon, a pop-up window will show up requiring the following information:

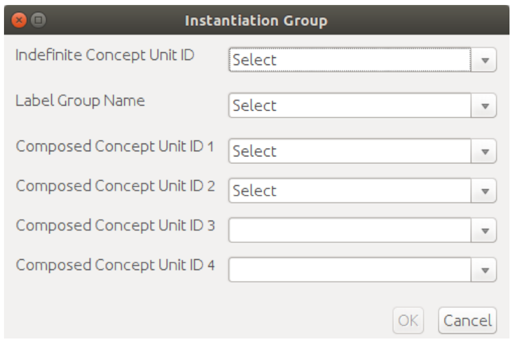


“Label Group Name” - used to specify the ID of the composed concept unit. Each ID must be unique. Underscore “_” should be used instead of spaces.

“Basic Concept Unit ID” - select one of the available options.

By clicking on the “Instantiation Groups Insert” icon, a pop-up window will show up requiring the following information:

“Indefinite Concept Unit ID” - select one of the available options. It must contain the indefinite concept “X” in its composition.


Instantiation Group
Indefinite Concept Unit ID
Select
Label Group Name
Select
Composed Concept Unit ID 1
Select
Composed Concept Unit ID 2
Select
Composed Concept Unit ID 3
Composed Concept Unit ID 4
OK
Cancel

“Label Group Name” - select one of the available options.

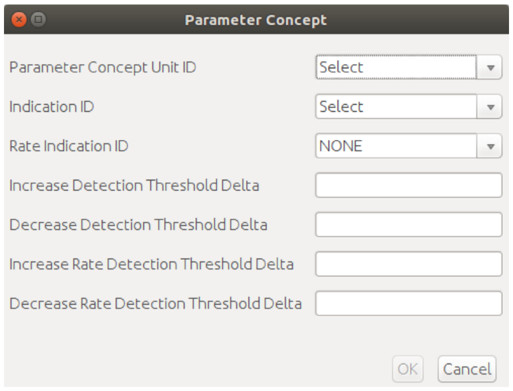
“Composed Concept Unit ID” - select one of the available options.

Step 19: Create Plant Crew Reasoning Machine Mental Representation

In the “Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Mental Representation Input” tab, the operator’s mental representation for knowledge-based reasoning is created.

The mental representation is used for bridging the operator’s ontology concept units with the external control room indicators. The indicators are classified into four types: parameter, component, support system, and alarm. The linkages between external indicators and operator’s ontology concepts are used for interpreting the perceived information from control panel and requesting information from control panel.

By clicking on the “Parameter Concepts Insert” icon, a pop-up window will show up requiring the following information:


Parameter Concept
Parameter Concept Unit ID
Select
Indication ID
Select
Rate Indication ID
NONE
Increase Detection Threshold Delta
Decrease Detection Threshold Delta
Increase Rate Detection Threshold Delta
Decrease Rate Detection Threshold Delta
OK
Cancel

“Parameter Concept Unit ID” - select one of the available options.

“Indication ID” - select one of the available options.

“Rate Indication ID” - select one of the available options.

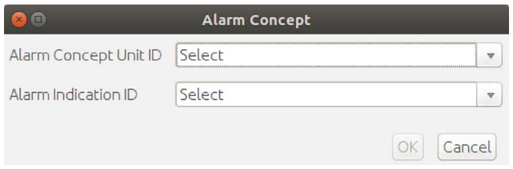
"Increase Detection Threshold Delta" - a positive value. If the newly perceived reading is greater than the detection threshold delta, the operator would detect that the parameter has increased in the interval.

"Decrease Detection Threshold Delta" - a negative value. If the newly perceived reading is smaller than the detection threshold delta, the operator would detect that the parameter has decreased in the interval.

"Increase Rate Detection Threshold Delta" – a positive value. If the newly perceived parameter changing rate is greater than the threshold, the operator would detect that the parameter is increasing.

"Decrease Rate Detection Threshold Delta" - a negative value. If the newly perceived parameter changing rate is less than the threshold, the operator would detect that the parameter is decreasing.

By clicking on the "Alarm Concepts Insert" icon, a pop-up window will show up requiring the following information:



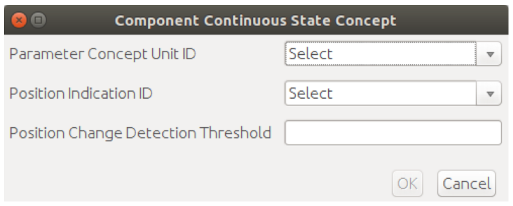
"Alarm Concept Unit ID" - select one of the available options.

"Alarm Indication ID" - select one of the available options.

Similarly, "Component Discrete State" and "Support System State Concepts" can be created.

By clicking on the "Component Continuous State Concepts Insert" icon, a pop-up window will show up requiring the following information:

“Parameter Concept Unit ID” - select one of the available options.

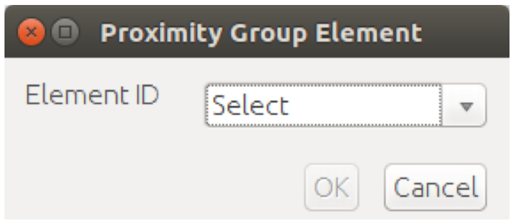

Component Continuous State Concept
Parameter Concept Unit ID
Select
Position Indication ID
Select
Position Change Detection Threshold
OK
Cancel

“Position Indication ID” - select one of the available options.

“Position Change Detection Threshold” - a positive value. The positive change threshold is of the same value as the negative change detection threshold. If the component position reading is greater or less than the last perceived reading and the difference is greater than the threshold, the operator would detect the position has been changed in this time interval.

Step 20: Create Plant Crew Reasoning Machine Concept Groups Base

In the “Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Concept Groups Base Input” tab, the operator’s proximity and antonym groups for knowledge-based reasoning are created.

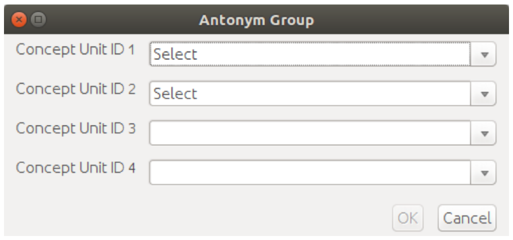


Proximity groups are used to associate elements in close proximity on the control panel and antonyms in groups. For example, if the operator sees Steam Generator A pressure increasing, they infer that “steam generator A pressure decreases” is a false statement of the current situation. Also, if the operator checks the Steam Generator A pressure, it also checks the Steam Generator B and C pressure indications.

By clicking the “Proximity Groups Insert” icon, a new proximity group is automatically created. By clicking on the “Corresponding Elements of Selected Proximity Groups Insert” icon, a pop-up window will show up requiring the following information:

"Element ID" - select one of the available options.

By clicking the "Antonym Groups Insert" icon, a pop-up window will show up requiring the following information:



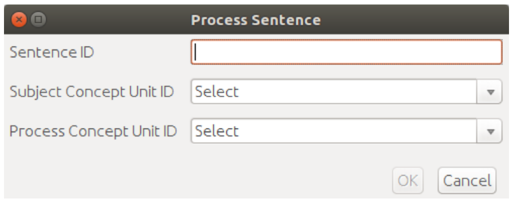
"Concept Unit ID" - select one of the available options.

Step 21: Create Plant Crew Reasoning Machine Semantic Base

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Semantic Base Input" tab, the operator's semantic base for knowledge-based reasoning is created.

It represents the operator's memory information in semantic sentences format. One semantic sentence describes a plant situational phenomenon - the plant/system/component state, parameter trend, or changes. The semantic sentences are built with basic concept units or composed concept units. There are five types of semantic sentences.

By clicking the "Process Sentences Insert" icon, a pop-up window will show up requiring the following information:



"Sentence ID" - ID of the sentence. Each ID must be unique. Underscore "_" should be used instead of spaces.

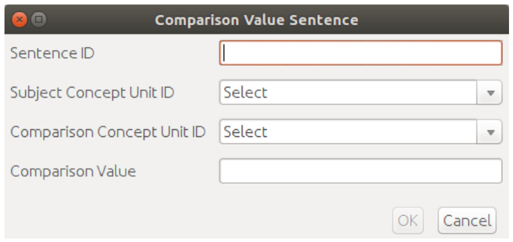
"Subject Concept Unit ID" - select one of the available options.

"Process Concept Unit ID" - select one of the available options.

Similarly, "Component State" and "Description Sentences" can be created.

By clicking the “Comparison Value Sentences Insert” icon, a pop-up window will show up requiring the following information:

“Sentence ID” - the ID of the sentence. Each ID must be unique. Underscore “_” should be used instead of spaces.

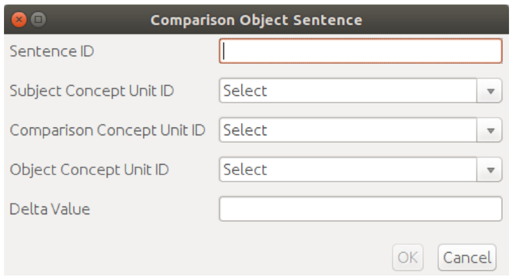


“Subject Concept Unit ID” - select one of the available options.

“Comparison Concept Unit ID” - select one of the available options.

“Comparison Value” - a constant value compared with the subject parameter/component position underlying the Comparison Concept Unit ID.

By clicking the “Comparison Object Sentences Insert” icon, a pop-up window will show up requiring the following information:



“Sentence ID” - the ID of the sentence. Each ID must be unique. Underscore “_” should be used instead of spaces.

“Subject Concept Unit ID” - select one of the available options.

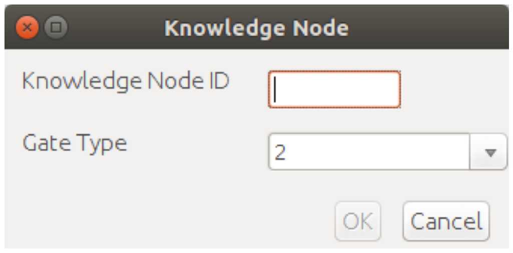
“Comparison Concept Unit ID” - select one of the available options.

“Object Concept Unit ID” - select one of the available options.

“Delta Value” - used to complete the comparison sentence. For example, if the Subject Concept Unit ID is “A”, the Comparison Concept Unit ID is “=” the Object Concept Unit ID is “B”, it is true when the difference between the perceived value of A and the perceived value of B is smaller than delta.

Step 22: Create Plant Crew Reasoning Machine Knowledge Base

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Knowledge Base Input" tab, the operator's knowledge base for knowledge-based reasoning is created.



Knowledge Node ID

Gate Type 2

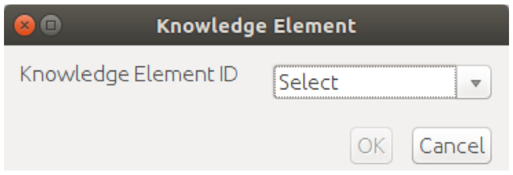
The operator's knowledge of the plant is represented in a knowledge web. This knowledge web is constructed by several nodes and associated links. A knowledge node contains more than one knowledge elements and combines them with a logic gate ("AND" gate or "OR" gate). A knowledge element has one semantic sentence and a negation flag ("TRUE" or "FALSE"). For each semantic sentence, two knowledge elements are automatically generated: one with "TRUE" flag and one with "FALSE" flag. The ID of the knowledge element with a "TRUE" flag is "KE_" + ID of its semantic sentence. The ID of the knowledge element with a "FALSE" flag is "KE_FALSE_" + ID of its semantic sentence. One knowledge node serves as one node in the knowledge web. One knowledge element could also serve as a node in the knowledge web.

By clicking the "Knowledge Nodes Insert" icon, a pop-up window will show up requiring the following information:

"Knowledge Node ID" - the ID of the knowledge node. Each ID must be unique. Underscore "_" should be used instead of spaces.

"Gate Type" - select "AND" or "OR."

By clicking the "Corresponding Knowledge Elements of Selected Knowledge Nodes Insert" icon, a pop-up window will show up requiring the following information:


Knowledge Element

Knowledge Element ID: Select

"Knowledge Element ID" - select one of the available options.

By clicking the "Knowledge Units Insert" icon, a pop-up window will show up requiring the following information:

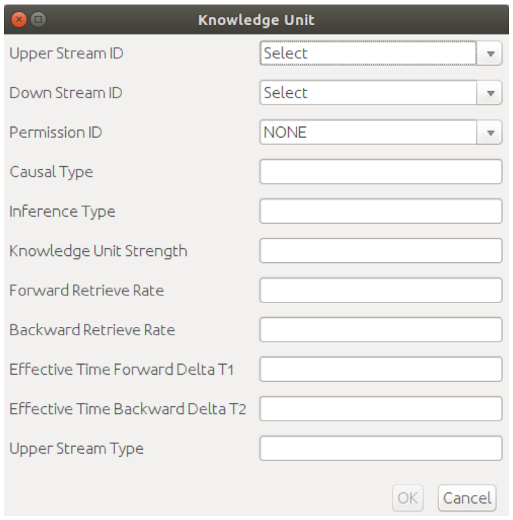
"Upper Stream ID" - select one of the available options.

"Down Stream ID" - select one of the available options.

"Permission ID" - select one of the available options.

"Causal Type" - select one of the available options.

"Inference Type" - select one of the available options.


Knowledge Unit

Upper Stream ID	Select
Down Stream ID	Select
Permission ID	NONE
Causal Type	
Inference Type	
Knowledge Unit Strength	
Forward Retrieve Rate	
Backward Retrieve Rate	
Effective Time Forward Delta T1	
Effective Time Backward Delta T2	
Upper Stream Type	

"Knowledge Unit Strength, Forward Retrieve Rate, Backward Retrieve Rate" – values between 0 and 1 that are used to evaluate the strength of each knowledge link. They represent the operator's ease or difficulty to recall that knowledge link given one end of this link is the retrieval cue. Small retrieval strength leads to longer recalling time or failure to retrieve that information.

"Effective Time Forward Delta T1 and Effective Time Backward Delta T2" - time delays of observing the downstream node phenomenon after the upstream phenomenon happened. "deltaT2" should be greater than "deltaT1." These two times are used to assess the expected time range for observing the downstream or upstream.

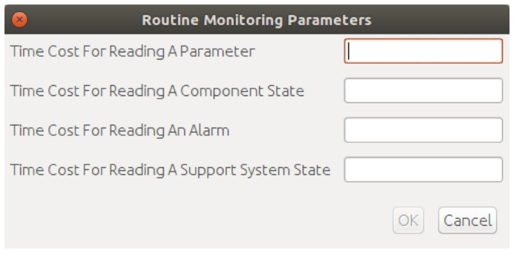
"Upper Stream Type" - select one of the available options.

Step 23: Create Plant Crew Reasoning Machine Routine Monitoring

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Routine Monitoring Input" tab, the operator's routine monitoring items and settings for knowledge-based reasoning are created.

It lists a set of indicators that are routinely monitored by the operators during the simulation. The user specifies the time interval between two consecutive monitoring during normal operation situation (no accident/no abnormal condition). By setting different time interval values for different indicators, the user could tailor the operator's routine style. Note that the time interval values in this input file will be only used a reference to calculate when the operator will do this routine monitor check of these indicators. They are not the exact time interval between two consecutive monitoring in the simulation. As operators need to react to the accident, the frequency of routine monitoring would be dynamically adjusted based on these values.

By clicking the "Routine Monitoring Settings Edit" icon, a pop-up window will show up requiring the following information:


Routine Monitoring Parameters

Time Cost For Reading A Parameter	
Time Cost For Reading A Component State	
Time Cost For Reading An Alarm	
Time Cost For Reading A Support System State	

"Time Cost For Reading A Parameter" – used to calculate how much time it costs the operator to read one parameter.

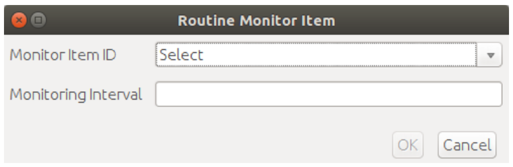
"Time Cost For Reading A Component State" - used to calculate how much time it costs the operator to read one component state.

"Time Cost For Reading An Alarm" - used to calculate how much time it costs the operator to read one alarm.

"Time Cost For Reading A Support System State" - used to calculate how much time it costs the operator to read one support system state.

By clicking the "Routine Monitoring Items Insert" icon, a pop-up window will show up requiring the following information:

"Monitor Item ID" - select one of the available options.



"Monitoring Interval" - time interval between two readings during normal operation. This is used as a base reference. During the simulation, the monitor time interval will be dynamically adjusted based on the context of the scenario.

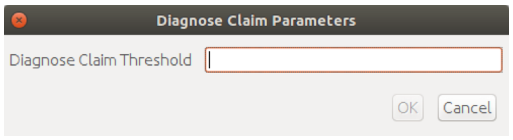
Step 24: Create Plant Crew Reasoning Machine Event Schema Base

In the "Plant Model -> Plant Crew -> Decision Maker -> Reasoning Machine -> Event Schema Base Input" tab, the operator's event schemas for knowledge-based reasoning are created.

An event schema is a structured unit that stores some pre-conceived typical pattern knowledge of an accident event. In addition to the knowledge web, it provides another way to organize and retrieve the operator's knowledge of an accident event. It highlights the knowledge causal paths from a root cause to one or more observable symptoms. It also links to a corresponding action procedure step for mitigating the accident. During the simulation, the event schema gets activated in the reasoning process when the operator doubts this accident might have happened. It becomes conscious to the operator when the confidence of this accident diagnosis is above a certain level. When the confidence exceeds a threshold, the operator will claim this accident has happened. Different levels of conservatism and rigorousness can be achieved by adjusting the threshold values.

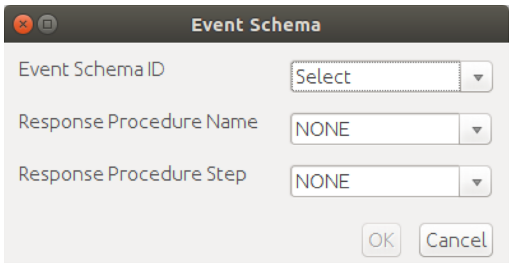
By clicking the "Diagnose Claim Threshold Edit" icon, a pop-up window will show up requiring the following information:

"Diagnose Claim Threshold" - the confidence threshold to claim an accident. During the simulation, the event schemas became active when the operator starts to see one or more symptoms and the symptoms trace back to this accident in the reasoning process. The program dynamically calculates the operator's confidence of this accident diagnosis. When the confidence increases above the threshold, the operator claims the accident has happened.



By clicking the "Event Schemas Insert" icon, a pop-up window will show up requiring the following information:

"Event Schema ID" - select one of the available options. It is the ID of the corresponding knowledge element that describes the accident event of this schema.

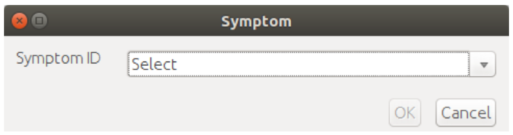


"Response Procedure Name" - select one of the available options. If none of the procedures apply to this event schema, "NONE" should be selected.

"Response Procedure Step" - select one of the available options. If none of the procedure steps apply to this event schema, "NONE" should be selected.

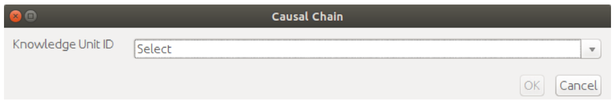
By clicking the "Corresponding Symptoms of Selected Event Schema Insert" icon, a pop-up window will show up requiring the following information:

"Symptom ID" - select one of the available options. It is the ID of the corresponding knowledge element that describes a symptom.



By clicking the "Corresponding Causal Chain of Selected Symptom" Insert icon, a pop-up window will show up requiring the following information:

"Knowledge Unit ID" - select one of the available options.



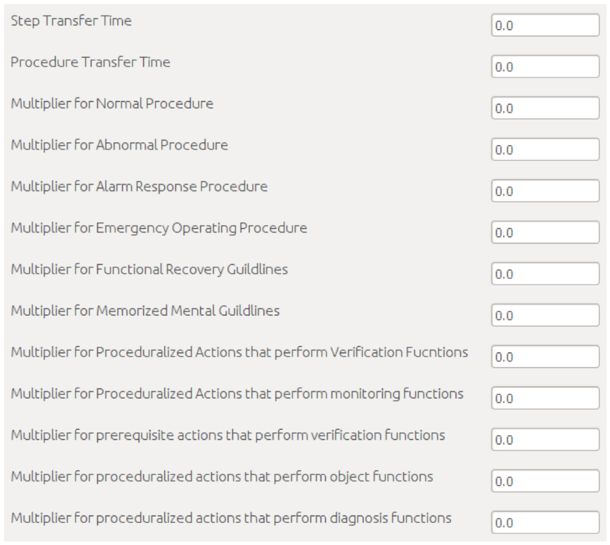
Step 25: Create Procedure Factors

In the "Plant Model -> Procedures -> Procedure Factors Input" tab, the operator's procedural characteristics for procedure-following strategy are created.

ADS-IDAC supports the modeling of omission of certain procedure actions in order model step skipping behavior. To provide adequate control over the simulation, step skipping behavior is limited to initial step actions and contingency "response not obtained" actions. The simulation approach requires that procedure steps be performed in sequence and that skipping behavior is applied at the sub-step level. If the steps within a procedure are subject to the same dependent factors, the model can generate sequences where all the steps of that procedure are skipped. The likelihood of skipping a sub-step is calculated by adjusting a base "skip step" probability by dynamic and static multipliers. These multipliers reflect procedural characteristics, the relevance of the action to the operator's situational assessment, and the PSF values. Procedure step skipping branches are generated to model the omissions of procedure actions based on the relevance of the step actions to the operator's situational assessment.

The skip multiplier falls within the range of 1.0 to 10.0 and reflects the crew's procedural adherence tendencies for various types of plant procedures. A higher value implies a greater likelihood of skipping the procedure step.

In this tab, the following information is required:



Step Transfer Time	0.0
Procedure Transfer Time	0.0
Multiplier for Normal Procedure	0.0
Multiplier for Abnormal Procedure	0.0
Multiplier for Alarm Response Procedure	0.0
Multiplier for Emergency Operating Procedure	0.0
Multiplier for Functional Recovery Guildlines	0.0
Multiplier for Memorized Mental Guildlines	0.0
Multiplier for Proceduralized Actions that perform Verification Fucntions	0.0
Multiplier for Proceduralized Actions that perform monitoring functions	0.0
Multiplier for prerequisite actions that perform verification functions	0.0
Multiplier for proceduralized actions that perform object functions	0.0
Multiplier for proceduralized actions that perform diagnosis functions	0.0

"Step Transfer Time" - nominal time delay to transition between steps within the same procedure.

"Procedure Transfer Time" - nominal time delay to transition between different procedures.

"Multiplier for Normal Procedure" - step skipping multiplier for normal operating procedures. Normal procedures typically include routine power changes and routine plant evolutions.

"Multiplier for Abnormal Procedure" - step skipping multiplier for abnormal operating procedures. Abnormal procedures are typically used to address non-routine events that do not constitute emergency or accident situations.

"Multiplier for Alarm Response Procedure" - step skipping multiplier for alarm response procedures. Alarm response procedures are used to guide operator follow up actions after a control panel alarm is activated.

"Multiplier for Emergency Operating Procedure" - step skipping multiplier for emergency EOPs.

“Multiplier for Functional Recovery Guidelines” - step skipping multiplier for functional recovery guidelines (FRGs). FRGs are used to address degradations of critical safety functions such as inventory control, core shutdown and cooling, and fission product containment.

“Multiplier for Memorized Mental Guidelines” - step skipping multiplier for memorized mental procedures. Memorized mental procedures are used to model skill-based actions carried out by the operators without reference to written procedures. These actions typically fall into the broad category of skill-of-the-craft activities.

“Multiplier for Proceduralized Actions that Perform Verification Functions” - skip multiplier for proceduralized actions that perform verification functions. Verification functions include checking the status of parameters and components where the operator does not normally expect to perform recovery actions.

“Multiplier for Proceduralized Actions that Perform Monitoring Functions” - skip multiplier for proceduralized actions that perform monitoring functions. Monitoring functions are generally associated with steps where an operator is required to observe the status of a parameter or component while performing other actions in parallel. Monitoring also includes observing the status of a changing parameter to initiate action when a threshold value is reached.

“Multiplier for Prerequisite Actions that Perform Verification Functions” - skip multiplier for proceduralized actions that perform verification functions. Prerequisite functions refer to actions that do not directly address the cause or symptoms of an ongoing event, but are needed to support later activities or prevent undesirable consequences of planned actions.

“Multiplier for Proceduralized Actions that Perform Object Functions” - skip multiplier for proceduralized actions that perform objective functions. Objective functions directly address the cause or symptoms of an ongoing event. These actions are usually central to the operator's understanding of the overall procedure goals and are less likely to be skipped.

"Multiplier for Proceduralized Actions that Perform Diagnosis Functions" - skip multiplier for proceduralized actions that perform diagnosis functions. Diagnosis functions involve the identification of the root cause(s) of an abnormal or emergency event. Diagnosis activities are generally focused on the identification of a specific failed component or system so that mitigating actions can be performed.

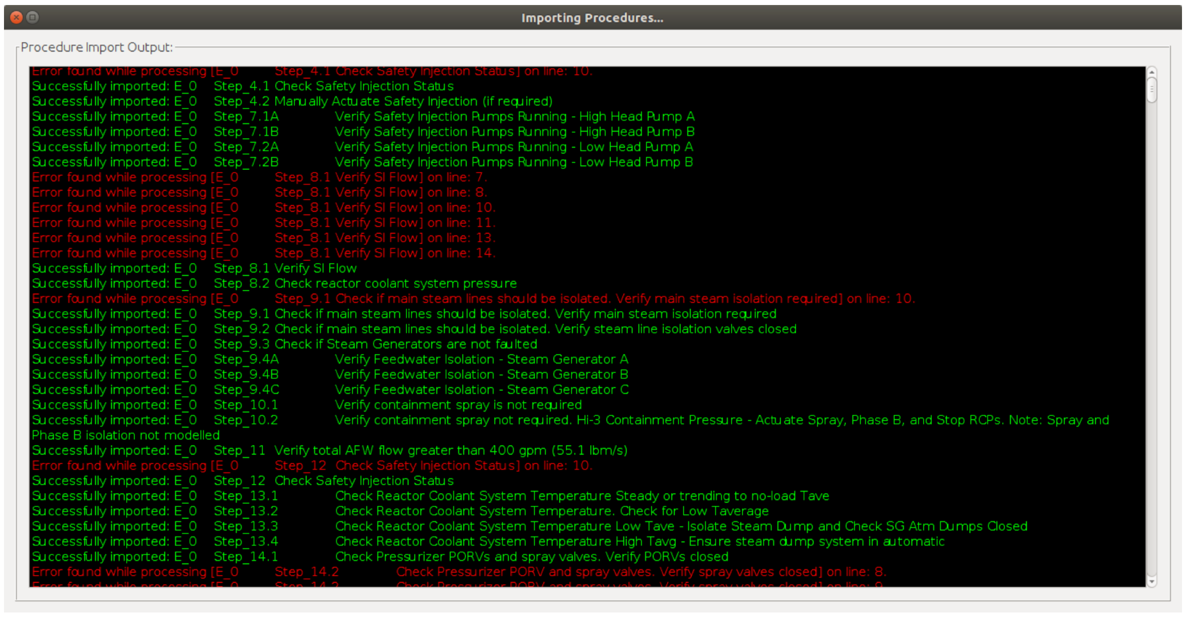
Step 26: Create Procedures and Steps

In the "Plant Model -> Procedures -> Procedures & Steps Input" tab, the operator's written procedures for procedure-following strategy are created.

ADS-IDAC includes the capability to represent both the structure and content of many types of plant procedures. Procedure step execution follows the standard format of action execution followed by expectation verification. If the action expectations are not met, a mitigating action can be performed. Four general types of procedural actions can be executed: (1) changing the component operating mode (e.g., automatic vs. manual mode), (2) setting a specific control value for a component (e.g., throttling control valve to 50% open), (3) incrementing the control setting of a component (e.g., throttling open a control valve by an additional 10%), and (4) setting a control value based on a perceived parameter (e.g., setting the steam dump target pressure equal to the perceived main steam header pressure). These capabilities provide sufficient flexibility to realistically model all significant operator interactions with the plant model.

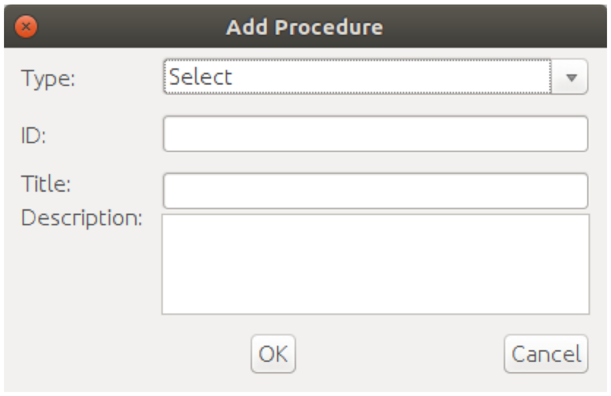
Generally, a written procedure is continued until the procedure is completed. The procedure flow may be interrupted by procedure transfers (which direct the crew to a different procedure), activation of an instinctive response action, or abandonment of the "Follow Written Procedure" strategy. Two types of procedure transfers can be modeled: (1) a permanent procedure transfer and (2) a temporary transfer to an auxiliary procedure followed by resumption of the initial procedure.

The task of creating the procedures is one of the most tedious. An importing and validating algorithm was implemented that greatly speeds up the process when previously written ADS-IDAC procedures were available. Errors during processing are highlighted in red, and the user manual should be used for correcting them.



```
Error found while processing [E_0       Step_4.1 Check Safety Injection Status] on line: 10.
Successfully imported: E_0    Step_4.1 Check Safety Injection Status
Successfully imported: E_0    Step_4.2 Manually Actuate Safety Injection (if required)
Successfully imported: E_0    Step_7.1A           Verify Safety Injection Pumps Running - High Head Pump A
Successfully imported: E_0    Step_7.1B           Verify Safety Injection Pumps Running - High Head Pump B
Successfully imported: E_0    Step_7.2A           Verify Safety Injection Pumps Running - Low Head Pump A
Successfully imported: E_0    Step_7.2B           Verify Safety Injection Pumps Running - Low Head Pump B
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 7.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 8.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 10.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 11.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 13.
Error found while processing [E_0       Step_8.1 Verify SI Flow] on line: 14.
Successfully imported: E_0    Step_8.1 Verify SI Flow
Successfully imported: E_0    Step_8.2 Check reactor coolant system pressure
Error found while processing [E_0       Step_9.1 Check if main steam lines should be isolated. Verify main steam isolation required] on line: 10.
Successfully imported: E_0    Step_9.1 Check if main steam lines should be isolated. Verify main steam isolation required
Successfully imported: E_0    Step_9.2 Check if main steam lines should be isolated. Verify steam line isolation valves closed
Successfully imported: E_0    Step_9.3 Check if Steam Generators are not faulted
Successfully imported: E_0    Step_9.4A           Verify Feedwater Isolation - Steam Generator A
Successfully imported: E_0    Step_9.4B           Verify Feedwater Isolation - Steam Generator B
Successfully imported: E_0    Step_9.4C           Verify Feedwater Isolation - Steam Generator C
Successfully imported: E_0    Step_10.1           Verify containment spray is not required
Successfully imported: E_0    Step_10.2           Verify containment spray not required. Hi-3 Containment Pressure - Actuate Spray, Phase B, and Stop RCPs. Note: Spray and
Phase B isolation not modelled
Successfully imported: E_0    Step_11  Verify total AFW flow greater than 400 gpm (55.1 lbm/s)
Error found while processing [E_0       Step_12  Check Safety Injection Status] on line: 10.
Successfully imported: E_0    Step_12  Check Safety Injection Status
Successfully imported: E_0    Step_13.1           Check Reactor Coolant System Temperature Steady or trending to no-load Tave
Successfully imported: E_0    Step_13.2           Check Reactor Coolant System Temperature. Check for Low Taverage
Successfully imported: E_0    Step_13.3           Check Reactor Coolant System Temperature Low Tave - Isolate Steam Dump and Check SG Atm Dumps Closed
Successfully imported: E_0    Step_13.4           Check Reactor Coolant System Temperature High Tavg - Ensure steam dump system in automatic
Successfully imported: E_0    Step_14.1           Check Pressurizer PORVs and spray valves. Verify PORVs closed
Error found while processing [E_0       Step_14.2           Check Pressurizer PORV and spray valves. Verify spray valves closed] on line: 8.
Error found while processing [E_0       Step_14.2           Check Pressurizer PORV and spray valves. Verify spray valves closed] on line: 9.
```

However, there are cases when new procedures need to be added. This can be achieved in the following way. By clicking the “New Procedure” icon, a pop-up window will show up requiring the following information:


Add Procedure

Type: Select

ID:

Title:

Description:

OK Cancel

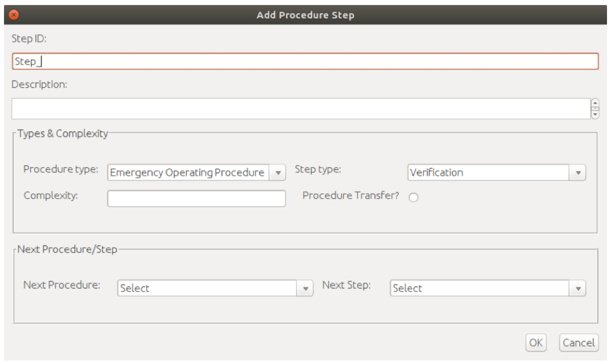
"Type" - select one of the available options.

"ID" - ID of the procedure name. Each ID must be unique. Underscore "_" should be used instead of spaces. Also, as mental procedures and functional recovery guidelines require special handling, the following procedure name prefixes are reserved to identify these procedure types: "FRG_" – Functional Recovery Guideline, "MPBG_" – Mental Procedure. It is also recommended (though not required) that the following ID prefixes be used: "ECA_" - Emergency Contingency Actions, "E_" - Emergency Operating Procedures, and "ES_" - Emergency Supplemental Procedure.

"Title" - title of the procedure (optional).

"Description" - more details regarding the procedure (optional).

By clicking the "Add Step" icon, a pop-up window will show up requiring the following information:


Add Procedure Step

Step ID:

Step_

Description:

Types & Complexity

Procedure type: Emergency Operating Procedure

Step type: Verification

Complexity:

Procedure Transfer?

Next Procedure/Step

Next Procedure: Select

Next Step: Select

OK Cancel

"Step ID" - ID of the procedure step. Each ID must be unique. Underscore "_" should be used instead of spaces.

"Description" - more details regarding the procedure step (optional).

"Procedure Type" - select one of the available options.

“Step Type” - select one of the available options.

“Complexity” - adjusts the step skipping probability. In general, it should be set to 1.0 (neutral). A value larger than 1.0 will increase the likelihood of skipping the step, while a value less than 1.0 will decrease the likelihood of skipping the step.

“Procedure Transfer” - tick the box if this is a procedure transfer.

“Next Procedure” - select one of the available options. Note, in some cases the next procedure may not have been created yet. Temporarily select “NONE” and after its addition return to select it.

“Next Step” - select one of the available options. Note, in some cases the next procedure step may not have been created yet. Temporarily select “NONE” and after its addition return to select it.

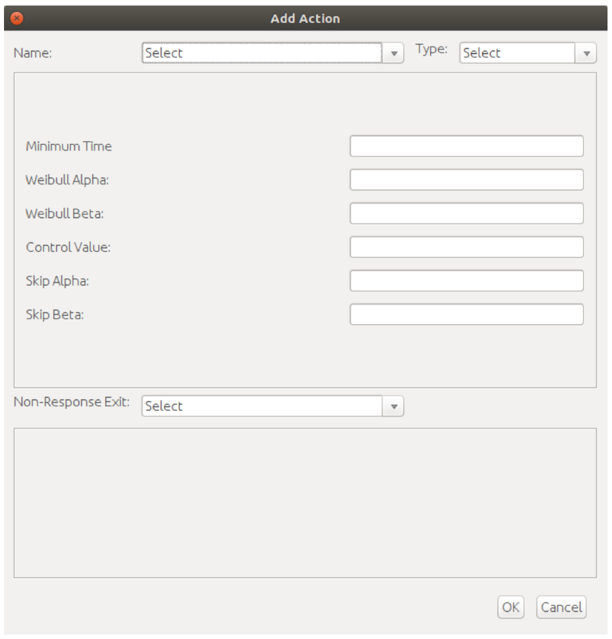
By clicking the “Actions Insert” icon, a pop-up window will show up requiring the following information:

“Name” - select one of the available options.

“Type” - select one of the available options.

“Minimum Time, Weibull Alpha, Weibull Beta” - used to calculate the time taken to perform the procedure step action, or perform non-response actions. Its uncertainty is modeled with a three parameter Weibull probability distribution.

“Control Value” - used to adjust the controller specified by the selected Name.



"Skip Alpha, Skip Beta" – parameters used to calculate the base procedure step skipping probability. Its uncertainty is modeled with the Beta distribution. This probability is adjusted during the simulation the static and dynamic factors defined at Step 25: Create Procedure Factors.

"Non-Response Exit" - select one of the available options.

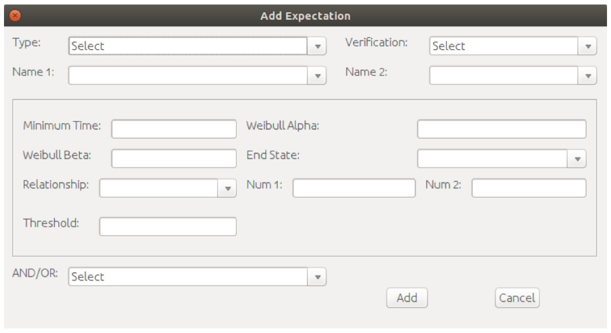
By clicking the "Action Expectations Insert" icon, a pop-up window will show up requiring the following information:

"Type" - select one of the available options

"Verification" - select one of the available options.

"Name 1, Name 2" - select one of the available options.

"Minimum Time, Weibull Alpha, Weibull Beta" - used to calculate the time taken to perform the action expectation. Its uncertainty is modeled with a three parameter Weibull probability distribution.



"End State" - select one of the available options.

"Relationship" - select one of the available options.

"Num 1, Num 2" - expected values for Name 1 and Name 2.

“Threshold” - used to evaluate the Name 1 and Name 2 difference.

“AND/OR” - select one of the available options. It is used to establish logical relationships between successive action expectations.

Step 27: Create Mental Procedure Activation Time

In the “Plant Model -> Procedures -> Procedure Activation Time Input” tab, the operator’s mental procedure activation times for procedure-following strategy are created.

It is one type of event sequence branch that can be generated during the procedure execution. After a mental belief is activated, the associated memorized mental procedure is initiated after the activation time delay has elapsed. Mental procedure activation time branches allow the analyst to examine the impact of variations in the activation time delay.

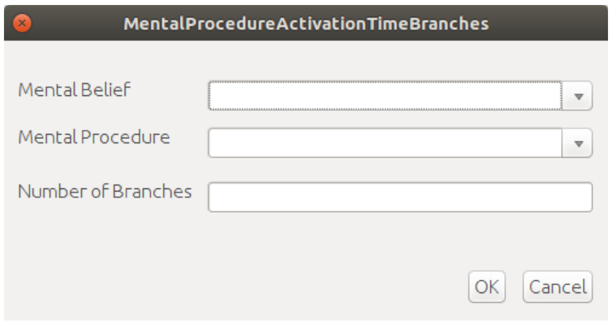
To capture the uncertainty and crew-to-crew variability associated with the time delay between activation of a mental belief and the execution of the associated memorized mental procedure, the activation delay is modeled with a three-parameter Weibull probability density distribution. When only one mental procedure activation time branch is generated, the activation time delay is equal to the mean value of the Weibull distribution. If more than one activation time delay branch is generated, the probability distribution is partitioned into one or more segments and the time delay for each sequence branch is determined by the mean value over the associated partition. The partition boundaries are determined by dividing the probability range of the Weibull cumulative probability distribution function into a number of segments equal to the number of desired branches.

By clicking the “Insert” icon, a pop-up window will show up requiring the following information:

“Mental Belief” - select one of the available options.

“Mental Procedure” - select one of the available options.

“Number of Branches” - the number of event sequence branches that will be generated.



Step 28: Create Procedure Action Time

In the “Plant Model -> Procedures -> Procedure Action Time Input” tab, the operator’s mental procedure action times for procedure-following strategy are created.

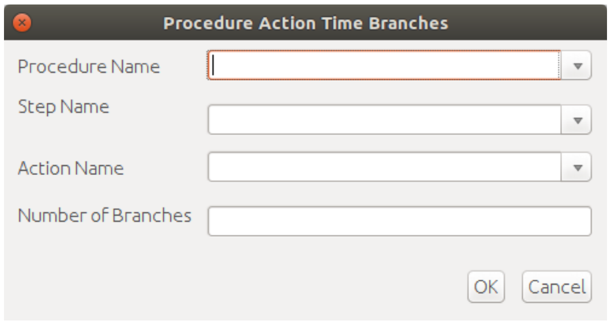
It is one type of event sequence branch that can be generated during the procedure execution. Action execution time branches enable multiple event sequence branches to be generated to model variations in the time taken by the control room crew in performing procedure actions. Its uncertainty and crew-to-crew variability are captured in the same way as for mental procedure activation times.

By clicking the “Insert” icon, a pop-up window will show up requiring the following information:

“Procedure name” - select one of the available options.

“Step name” - select one of the available options.

“Action Name” - select one of the available options.



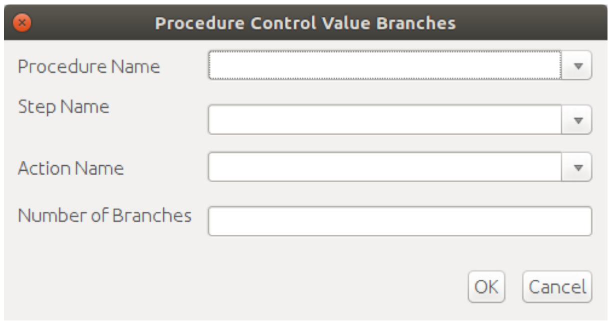
"Number of Branches" - the number of event sequence branches that will be generated.

Step 29: Create Procedure Control Value

In the "Plant Model -> Procedures -> Procedure Control Value Input" tab, the operator's mental procedure control values for procedure-following strategy are created.

It is one type of event sequence branch that can be generated during the procedure execution. Action control value branches can be used to model variations in control inputs such as control valve positioning and the setting of control system target set points.

By clicking the "Procedure Control Value Branches Insert" icon, a pop-up window will show up requiring the following information:



"Procedure name" - select one of the available options.

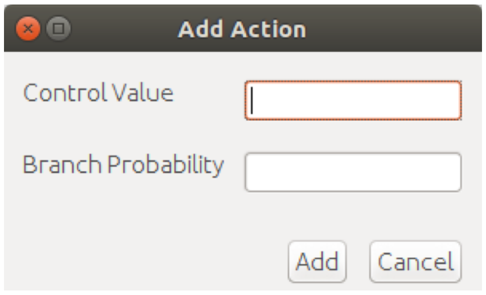
"Step name" - select one of the available options.

"Action Name" - select one of the available options.

"Number of Branches" - the number of event sequence branches that will be generated.

By clicking the "Control Branch Insert" icon, a pop-up window will show up requiring the following information:

"Control Value" - appropriate for the associate controller defined at "Action Name."

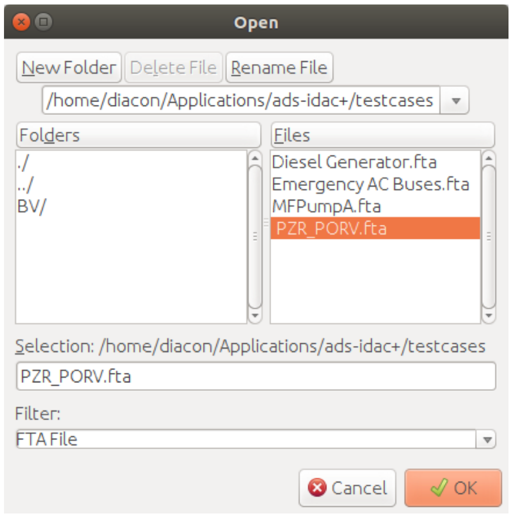


"Branch Probability" - between 0.0 and 1.0. The sum of all branch probabilities for this procedure control value branching rule should sum to 1.0.

Step 30: Import Frontline Systems Fault Trees

In the "Plant Model -> Hardware Reliability -> Frontline System Fault Trees Input" tab, the plant's frontline system fault tree *.fta files are imported. These are *.xml formatted files that can be generated with the Trilith software.

By clicking on the "Import" icon, a pop-up window will show up requiring the selection of an *.fta file.



/home/diacon/Applications/ads-idac+/testcases

./
../
BV/

Diesel Generator.fta
Emergency AC Buses.fta
MFPumpA.fta
PZR_PORV.fta

Selection: /home/diacon/Applications/ads-idac+/testcases

PZR_PORV.fta

FTA File

Step 31: Create System Reliability Parameters

In the "Plant Model -> Hardware Reliability -> System Reliability Parameters Input" tab, the plant's system reliability parameters are created.

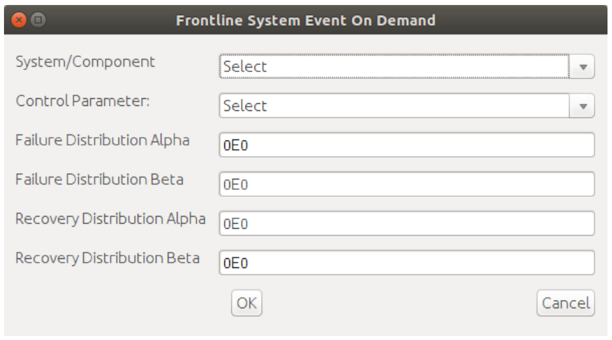
ADS-IDAC has great flexibility in accommodating a wide range of hardware failures for ASP analysis. System failures can occur on demand, during operation or at fixed times. It is important to note that the first two are time independent: they activate only when the specified component is needed. For on demand failures, two branches are generated:

A "failure branch." For this branch, the operator may attempt to manually recover the failed component through appropriate actions. If a recovery is attempted, two new branches are generated: a "successful recovery branch" and a "permanent failure branch."

"A success branch." For this branch, failures during operation may occur at specified time intervals after the initial time of demand. Two more branches are generated: "success" and "failure" branches. If more than one failure during operation for a specific component is implemented, only the success branches will experience further failures. Recovery is allowed if the operators manually attempt appropriate actions.

For failures at fixed times, a single failure branch is generated that also allows for the operators to attempt recovery actions.

By clicking the "System Failures On Demand Insert" icon, a pop-up window will show up requiring the following information:


Frontline System Event On Demand

System/Component	Select
Control Parameter:	Select
Failure Distribution Alpha	0E0
Failure Distribution Beta	0E0
Recovery Distribution Alpha	0E0
Recovery Distribution Beta	0E0

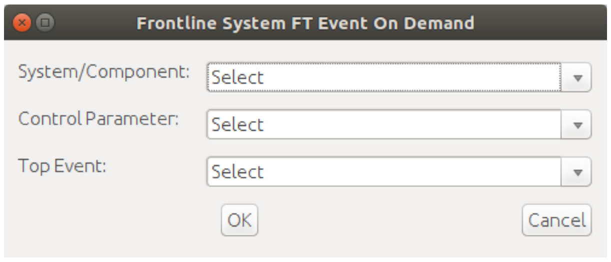
"System/Component" - select one of the available options.

"Control Parameter" - select one of the available options.

"Failure Distribution Alpha, Beta" – beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

"Recovery Distribution Alpha, Beta" – beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

By clicking the “System Failures On Demand With Fault Tree Insert” icon, a pop-up window will show up requiring the following information:


Frontline System FT Event On Demand

System/Component: Select

Control Parameter: Select

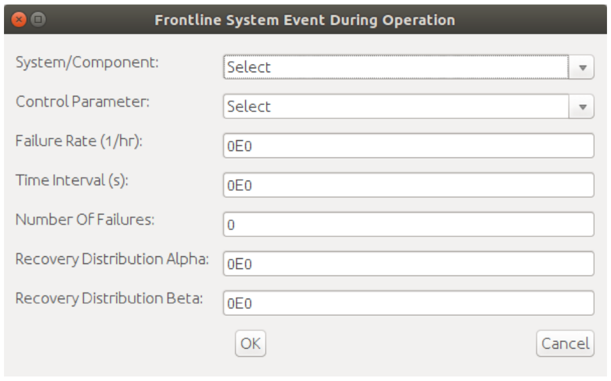
Top Event: Select

“System/Component” - select one of the available options.

“Control Parameter” - select one of the available options.

“Top Event” - select the top event of the FT associated with the selected system. At each demand, the top event probability is recalculated.

By clicking the “System Failures During Operation Insert” icon, a pop-up window will show up requiring the following information:


Frontline System Event During Operation

System/Component: Select

Control Parameter: Select

Failure Rate (1/hr): 0E0

Time Interval (s): 0E0

Number Of Failures: 0

Recovery Distribution Alpha: 0E0

Recovery Distribution Beta: 0E0

“System/Component” - select one of the available options.

“Control Parameter” - select one of the available options.

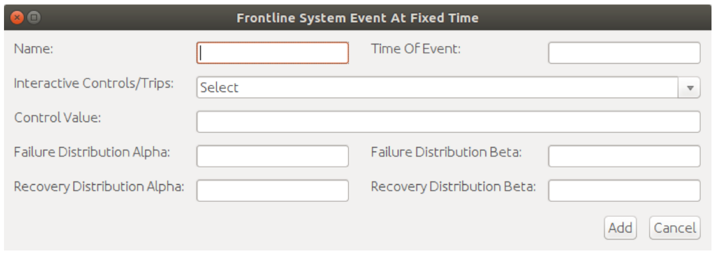
“Failure Rate (1/hr)” – failure rate.

“Time Interval (s)” - time interval after the initial demand and between successive failure events.

“Number of Failures” - number of failures that can occur during operation.

“Recovery Distribution Alpha, Beta” – beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

By clicking the “System Failures At Fixed Times Insert” icon, a pop-up window will show up requiring the following information:


Frontline System Event At Fixed Time
Name:
Time Of Event:
Interactive Controls/Trips:
Select
Control Value:
Failure Distribution Alpha:
Failure Distribution Beta:
Recovery Distribution Alpha:
Recovery Distribution Beta:
Add
Cancel

“Name” - name for this event. Underscore “_” should be used instead of spaces.

“Time Of Event” - the simulation time when this event will be activated.

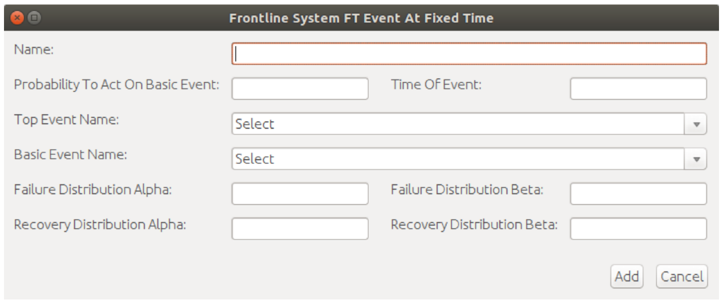
“Interactive Controls/Trips” - select one of the available options.

“Control Value” - control value for the selected interactive control.

“Failure Distribution Alpha, Beta” - beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

“Recovery Distribution Alpha, Beta” - beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

By clicking the “System Failures At Fixed Times With Fault Tree Insert” icon, a pop-up window will show up requiring the following information:


Frontline System FT Event At Fixed Time
Name:
Probability To Act On Basic Event:
Time Of Event:
Top Event Name:
Select
Basic Event Name:
Select
Failure Distribution Alpha:
Failure Distribution Beta:
Recovery Distribution Alpha:
Recovery Distribution Beta:
Add
Cancel

“Name” - name for this event. Underscore “_” should be used instead of spaces.

“Probability to Act on Basic Event” - new basic event probability.

“Time Of Event” - simulation time when this event will be activated.

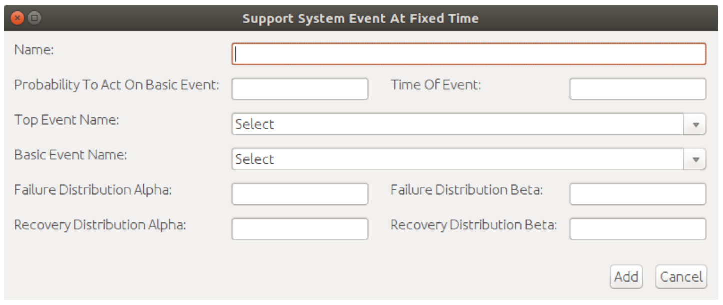
“Top Event Name” - select one of the available options.

“Basic Event Name” - select one of the available options.

“Failure Distribution Alpha, Beta” - beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

“Recovery Distribution Alpha, Beta” - beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

By clicking the “Support System Failures At Fixed Times Insert” icon, a pop-up window will show up requiring the following information:


Support System Event At Fixed Time
Name:
Probability To Act On Basic Event:
Time Of Event:
Top Event Name:
Select
Basic Event Name:
Select
Failure Distribution Alpha:
Failure Distribution Beta:
Recovery Distribution Alpha:
Recovery Distribution Beta:
Add
Cancel

“Name” - name for this event. Underscore “_” should be used instead of spaces.

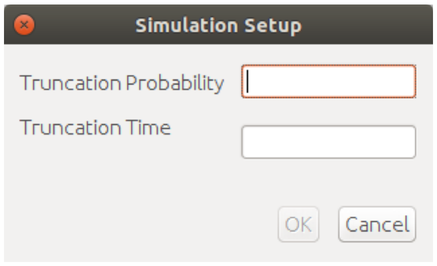
“Probability to Act on Basic Event” - new basic event probability.

“Time Of Event” - simulation time when this event will be activated.

“Top Event Name” - select one of the available options.

“Basic Event Name” - select one of the available options.

“Failure Distribution Alpha, Beta” – beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.



“Recovery Distribution Alpha, Beta” - beta probability distribution parameters. To capture the uncertainty associated with failure estimates, noting that at this moment only the mean is used.

Step 32: Create Simulation Setup

In the “Simulation Input” tab, the simulation setup settings for critical simulation control are set.

By clicking the “Edit” icon, a pop-up window will show up requiring the following information:

“Truncation Probability” - the probability cutoff. If the sequence probability is less than the truncation probability, the sequence is stopped.

“Truncation Time” - time cutoff. If the sequence simulation time is larger than the truncation time, the sequence is stopped.

References

Aldemir, T. (2013). A survey of dynamic methodologies for probabilistic safety assessment of nuclear power plants. *Annals of Nuclear Energy*, *52*, 113–124. http://doi.org/10.1016/j.anucene.2012.08.001

Antonov, I. A., and V. M. Saleev, (1979). An economic method of computing LPτ-sequences. USSR Comput. Math. Math. Phys., 19, no. 1, pp. 252–256.

Azarkhil, M. (2013). Dynamic Behavior of Operating Crew in Complex Systems. University of Maryland.

Beachkofski, B. and R. Grandhi, (2002) Improved Distributed Hypercube Sampling, in 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, American Institute of Aeronautics and Astronautics.

Cacciabue, P. C. (1992). Cognitive modeling: a fundamental issue for human reliability assessment methodology? Reliability Engineering & System Safety, vol. 38, pp. 91-97

Chang, Y.-H. (2007). Cognitive Modeling and Dynamic Probabilistic Simulation of Operating Crew Response to Complex System Accidents (ADS-IDA Crew). University of Maryland.

Chang, Y. H. and A. Mosleh. (2007). Cognitive modeling and dynamic probabilistic simulation of operating crew response to complex system accidents - Part 1: Overview of the IDAC Model, *Reliability Engineering & System Safety*, vol. 92, pp. 997-1013

Chang, Y.H., D. Bley, L. Criscione, B. Kirwan, A. Mosleh, T. Madary, R. Nowell, R. Richards, E. M. Roth, S. Sieben, A. Zoulis. (2014) The SACADA database for human reliability and human performance, *Reliability Engineering & System Safety*, Volume 125, Pages 117-133, ISSN 0951-8320, https://doi.org/10.1016/j.ress.2013.07.014.

Cheok, M. (2004). Accident Sequence Precursor Program. Washington, D.C.: 4th NRC – AERB Nuclear Safety Projects Meeting.

Coyne, K. A. (2009). A Predictive Model of Nuclear Power Plant Crew Decision-Making And Performance In A Dynamic Simulation Environment. University of Maryland.

Coyne, K. A. and Siu, N. (2013). Simulation-Based Analysis for Nuclear Power Plant Risk Assessment: Opportunities and Challenges. Washington, D.C: Proceeding of the ANS Embedded Conference on Risk Management for Complex Socio-Technical Systems.

Devooght, J., Smidts, C. (1996). Probabilistic dynamics as a tool for dynamic PSA. Reliability Engineering and System Safety, 52(3 SPEC. ISS.), 185–196. http://doi.org/10.1016/0951-8320(95)00135-2.

Ekanem, N. J. (2013). A Model-Based Human Reliability Analysis Methodology (Phoenix Method). University of Maryland.

Faure, H. (1982). Discrépance de suites associées à un système de numération (en dimension s), Acta Arith., vol. 41, no. 4, pp. 337–351.

Garrick, B. J. (2013) PRA based risk management: history and perspectives. *Closing Plenary Session at American Nuclear Society Winter Meeting*.

Groen, E. http://evelynegroen.github.io/Code/Uncertaintypropagation.html Accessed on August 2017.

Groen, F.J., C. Smidts, A. Mosleh. (2006). Qras - the Quantitative Risk Assessment System. Reliability Engineering & System Safety, 91, 3: p. 292-304.

Final Precursor Analysis of H.B. Robinson Electrical Fault Causes Fire and Subsequent Reactor Trip with a Loss of Rcp Seal Injection and Cooling. 2010, US NRC.

Halton, J.H. (1960). On the Efficiency of Certain Quasi-random Sequences of Points in Evaluating Multi-dimensional Integrals, Numer Math, vol. 2, no. 1, pp. 84–90.

Hammersley, J. M. (1960) Monte Carlo Methods for Solving Multivariable Problems, Ann. N. Y. Acad. Sci., vol. 86, no. 3, pp. 844–874.

Henrion, M. (1989). Some practical issues in constructing belief networks. Uncertainty in Artificial Intelligence, Elsevier Science Publishers, pp. 161-173

Kanal, L.N., Levitt, T.S., Lemmer, J.F. (eds.), Uncertainty in Artificial Intelligence 3. Elsevier Science Publishers B.V. (North Holland), pages 161-173

Hollnagel, E. (1998). Cognitive Reliability and Error Analysis Method (CREAM). Cognitive Reliability and Error Analysis Method (CREAM). Elsevier. http://doi.org/10.1016/B978-008042848-2/50006-3

Hsueh, K. S., A. Mosleh. (1996). The development and application of the accident dynamic simulator for dynamic probabilistic risk assessment of nuclear power plants, Reliability Engineering & System Safety, vol. 52, pp. 297-314.

Judea, P. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc., San Mateo, CA.

Kaplan, S. and Garrick, B. J. (1981). On the Quantitative Definition of Risk. Risk Analysis, 1(1), 11–27. http://doi.org/10.1007/BF00134104

Kauffman, J. V. (1995). Engineering Evaluation - Operating Events With Inappropriate Bypass or Defeat of Engineered Safety Features. US Nuclear Regulatory Commission, Office of Analysis and Evaluation of Operational Data AEOD/E95-01.

Koiter, Joost R. (2006). Visualizing Inference in Bayesian Networks. M.Sc. thesis, Faculty of Electrical Engineering, Mathematics, and Computer Science, Department of Man-Machine Interaction, Delft University of Technology

Klein, G. (1997). The Recognition-Primed Decision (RPD) Model: Looking Back, Looking Forward. Naturalistic Decision Making. pp. 285-292.

Li, Y. (2013). Modeling and Simulation of Operator Knowledge-Based Behavior. University of Maryland.

Matsumoto, M. and T. Nishimura. (1998). Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. TOMACS, vol. 8, no. 1, pp. 3–30.

Mosleh, A. (2014). PRA: A Perspective on strengths, current Limitations, and possible improvements. Nuclear Engineering and Technology, 46(1), 1–10. http://doi.org/10.5516/NET.03.2014.700

Mosleh, A. and Y. H. Chang, (2004) Model-based human reliability analysis: prospects and requirements," Reliability Engineering and System Safety, vol. 83, pp. 241-253

Niederreiter, H. (1988). Low-discrepancy and low-dispersion sequences," J. Number Theory, vol. 30, no. 1, pp. 51–70.

Niederreiter, H. (1987) Point sets and sequences with small discrepancy," Monatshefte Für Math., vol. 104, no. 4, pp. 273–337.

Pearl, J. (1988) Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann.

Rasmussen, N. (1975). Reactor Safety Study: An assessment of Accident Risks in U.S. Commercial Nuclear Power Plants WASH 1400 (NUREG-75/014).

Rauzy, A. and Y. Dutuit. (1997). Exact and Truncated Computations of Prime Implicants of Coherent and Non-Coherent Fault Trees within Aralia. Reliability Engineering & System Safety, 58, 2: p. 127-144.

Sanders, G. (2010) Licensee Event Report No. 2010- 002-00, Plant Trip Due to Electrical Fault, H. B. Robinson Steam Electric Plant, Unit No. 2.

Siu, N. (1994). Risk assessment for dynamic systems: An overview. Reliability Engineering & System Safety, 43(1), 43–73. http://doi.org/10.1016/0951-8320(94)90095-7

Siu, N. (1990). Dynamic accident sequence analysis in PRA: A comment on “Human reliability analysis-Where shoudst thou turn?” Reliability Engineering and System Safety, 29(3), 359–364. http://doi.org/10.1016/0951-8320(90)90019-J

Smidts, C., S.-H. Shen, and A. Mosleh. (1995). A Taxonomy And Root-Cause Analysis Of Human Cognitive Behavior Based On A Cognitive Model, presented at Annual Reliability and Maintainability Symposium,

Smidts, C., S. H. Shen, and A. Mosleh (1997) The IDA cognitive model for the analysis of nuclear power plant operator response under accident conditions. Part I: Problem solving and decision making model," Reliability Engineering & System Safety, vol. 55, pp. 51-71.

Swain, A. D., Guttmann, H. E. (1983). Handbook of human reliability analysis with emphasis on nuclear power plant applications: Final report. Washington, D.C.

The RELAP5 Development Team. (1995). RELAP5/M0D3 Code Manual NUREG/CR-5535.

van der Corput, J., (1935). Verteilungsfunktionen. i. mitt, in Proc. Akad. Wet. Amsterdam, , vol. 38, pp. 813–821.

Wang, C. (2007). Hybrid Causal Logic Methodology for Risk Assessment. University of Maryland.