SAND2019-12015

Light Water Reactor Sustainability Program

Modeling for Existing Nuclear Power Plant Security Regime

October 2019

U.S. Department of Energy

Office of Nuclear Energy

DISCLAIMER

This information was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness, of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the U.S. Government or any agency thereof.

SAND2019-12015

Modeling for Existing Nuclear Power Plant Security Regime

Douglas Osborn, Brain Cohn, M. Jordan Parks, Ryan Knudsen, Kyle Ross, Chris Faucett, Troy Haskin, Peter Kitsos, and Todd Noel

October 2019

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy

(This Page Intentionally Left Blank)

ABSTRACT

This document details the development of modeling and simulations for existing plant security regimes using identified target sets to link dynamic assessment methodologies by leveraging reactor system level modeling with force-on-force modeling and 3D visualization for developing table-top scenarios. This work leverages an existing hypothetical example used for international physical security training, the Lone Pine nuclear power plant facility for target sets and modeling.

(This Page Intentionally Left Blank)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

2S	Safety-Security
DBT	Design Basis Threat
BWR	Boiling Water Reactor
CAS	Central Alarm Station
CFD	Computational fluid Dynamic
CST	Condensate Storage Tank
DET	Dynamic Event Tree
DOE	U.S. Department of Energy
DOE-NE	U.S. Department of Energy’s Office of Nuclear Energy
EPRI	Electric Power Research Institute
IAEA	International Atomic Energy Agency
INL	Idaho National Laboratory
INMM	Institute of Nuclear Material Management
INPO	Institute of Nuclear Power Operations
ITC	International Training Course
LHF	Lower Head Failure
LPNPP	Lone Pine Nuclear Power Plant
LWR	Light Water Reactor
LWRS	Light Water Reactor Sustainability
MSL	Main Steam Line
NEI	Nuclear Energy Institute
NNSA	National Nuclear Security Administration
NPP	Nuclear Power Plant
NRC	U.S. Nuclear Regulatory Commission
PRA	Probabilistic Risk Assessment
PRT	Pressurizer Relief Tank
PWR	Pressurized Water Reactor
RCIC	Reactor Core Isolation Cooling
RCP	Reactor Coolant Pump
RCS	Reactor Coolant System
RN	Radionuclide
RPV	Reactor Pressure Vessel
SAS	Secondary Alarm Station
SBO	Station Blackout
SG	Steam Generator
SNL	Sandia National Laboratories
SOARCA	State-of-the-Art Reactor Consequence Analysis
STSBO	Short-Term Station Blackout
SV	Safety Valve
TDAFW	Turbine Driven Auxiliary Feedwater
TMI-2	Three Mile Island Unit 2

1. Introduction

This report documents the Sandia National Laboratories (SNL) efforts for evaluating a linked dynamic assessment framework for force-on-force modeling and reactor system response modeling. This report highlights the modeling work completed to date and the current challenges associated linking dynamic models.

The ultimate goal of this work is to develop a more realistic basis for modeling and simulations of an existing plant's security regime. This work leverages an existing hypothetical example used for international physical security training, the Lone Pine nuclear power plant facility for target sets and modeling.

1.1 Motivation

Domestic nuclear power generation faces increasing economic pressures, in part, by post-Fukushima regulatory requirements, an increase in subsidized renewable energy sources, and current low-cost natural gas. The requirements for U.S. nuclear power generation sites, post-9/11, to maintain a large on-site physical security force ranks high for related plant operational costs; ~12% of the overall cost (~$560 million) for decommissioning a nuclear facility [1]. U.S. nuclear power plants are seeking novel physical security methods and technologies to help deliver on the Nuclear Promise [2].

DOE National Laboratories have extensively studied physical security configurations that couple detect, delay, and response attributes to regulatory required physical security postures. This DOE Office of Nuclear Energy (DOE-NE) Light Water Sustainability (LWRS) Program effort seeks to create tools, methods, and technologies that will:

- Apply aspects of risk-informed techniques for physical security decisions and activities to account for a dynamic adversary;

- Apply advanced modeling and simulation tools to better inform physical security posture;

- Assess benefits from proposed enhancements, novel mitigation strategies, and potential changes to regulations; and

- Enhance the technical basis necessary for operating utilities to reevaluate their physical security posture while meeting regulatory requirements.

2. Lone Pine Nuclear Power Plant Overview

The Lone Pine Nuclear Power Plant (LPNPP) is a two-loop pressurized water reactor (PWR) with a reactor power level of 1150 megawatts electric at full power. The system consists of a reactor, a closed primary coolant loop connected to the reactor vessel, and a closed separate power conversion system (secondary coolant) for the generation of steam to power the turbine(s). Cooling water to the main condenser is provided from the river. Full details of the LPNPP can be found in Reference [3].

The primary coolant is light water under pressure (typically 2235 psi) containing chemicals to control the nuclear reaction (boric acid, referred to as “chemical shim”) and corrosion. The secondary coolant is also light water containing chemicals to control corrosion. The primary coolant system transfers heat from the reactor core to the steam generators, which transfer heat to the secondary coolant, causing it to boil. The steam passes from the steam generators to the turbine generator where the thermal energy of the steam is converted into mechanical and then electrical energy. The steam is condensed in the main condenser, and the secondary coolant is returned to the steam generators by the feedwater pumps. The use of a dual cycle minimizes the quantities of fission products released to the main turbine, condenser, and other secondary plant components, and subsequent release to the atmosphere.

The entire Reactor Coolant System (RCS), including the steam generators, is located in the Containment Building, which isolates the radioactive RCS from the environment in the event of a leak. The basic arrangement is shown in Figure 2.1. Additionally, the LPNPP is a once-through tertiary cooling system with a cooling tower and the river supplies the ultimate heat sink.

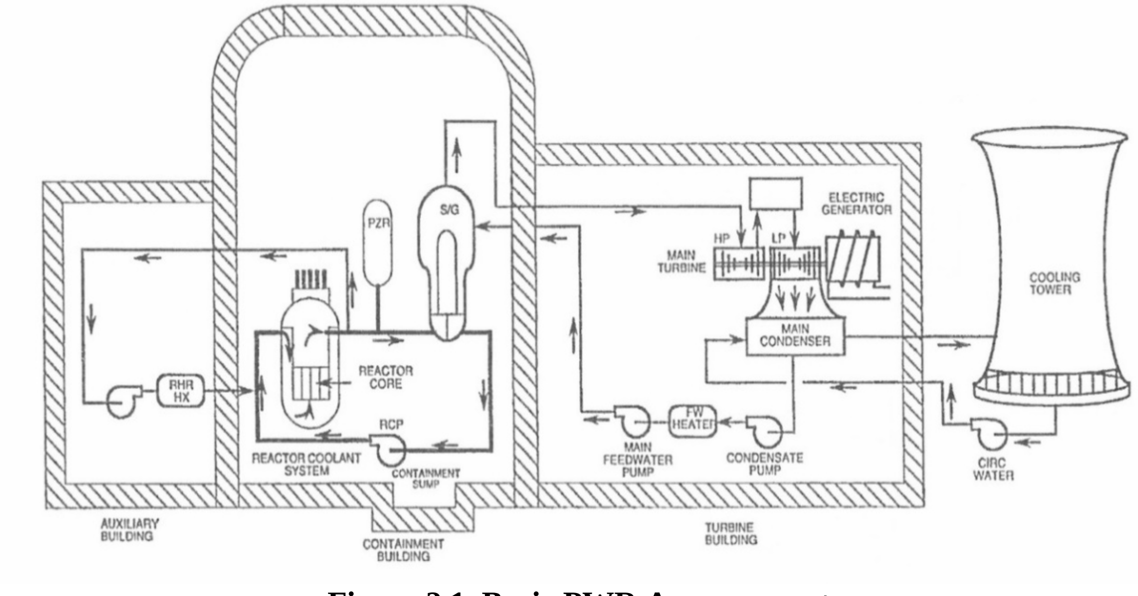


Figure 2.1 Basic PWR Arrangement

The following major systems are included in the water intake structure:

- **Circulating Water System.** The six pumps of this system take water from the river and provide cooling to the main condenser. This cooling water is then discharged via a cooling canal to the river.
- **Service Water System.** The four pumps of this system also take water from the river and discharge via the cooling canal to the river. Service water is used to cool other systems, such as the primary and secondary (steam) system, component cooling system, containment cooler, diesel generators and other heat exchangers.
- **Screen Wash System.** Six traveling water screens are provided to remove trash and foreign mater from the water used to supply the service and circulating water systems.

2.1 LPNPP Overall Site Layout

The overall LPNPP site layout is shown in Figure 2.2. Additionally, Figure 2.2 shows the locations of the guard posts, central alarm station (CAS), guard towers, and secondary alarm station (SAS).

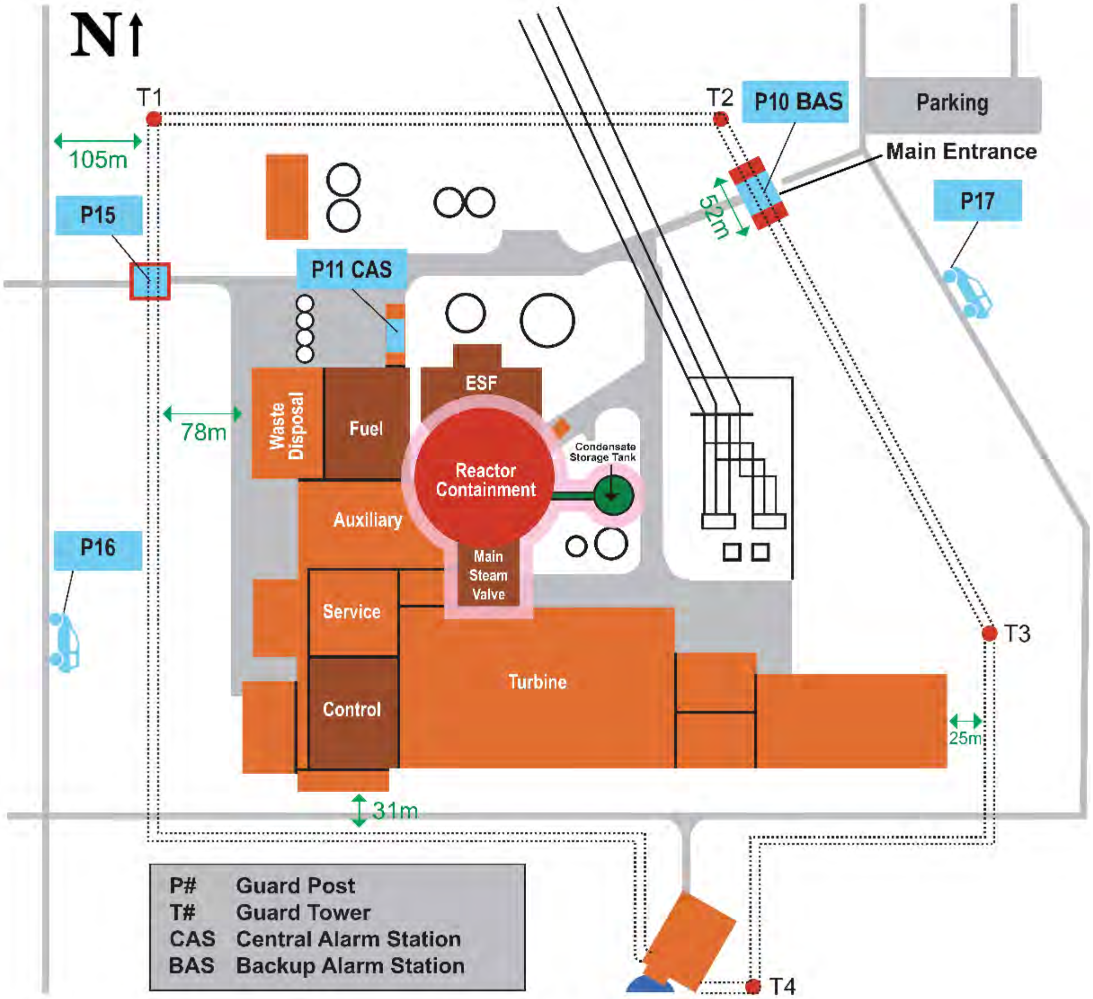


Figure 2.2 LPNPP Site Layout and Guard Posts

2.2 History of LPNPP Modeling

In 2011 the LPNPP was first created as a hypothetical model for use in training courses on physical security, vital area identification, plant safety, and other topics. Additionally, the model consisted of a plant description document with drawings, artists' renderings, and plant layouts as shown in Figure 2.3.

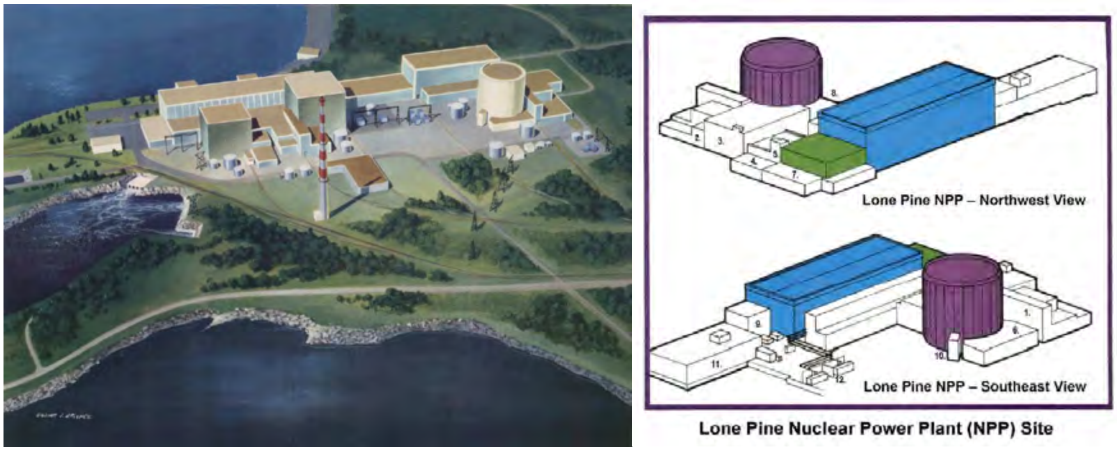

Lone Pine NPP – Northwest View
Lone Pine NPP – Southeast View
Lone Pine Nuclear Power Plant (NPP) Site

Figure 2.3 LPNPP 2011 Artist Rendering

In 2013, Lone Pine was first modelled in 3D for facility flythrough and security analysis scenarios. The model was created in Presagis Creator software (shown in Figure 2.4), eventually moving to Blender for use in the Unity gaming engine. The 2013 model featured very little in terms of building interiors and had no detail with regards to reactor containment. The only interior facility was the control room.

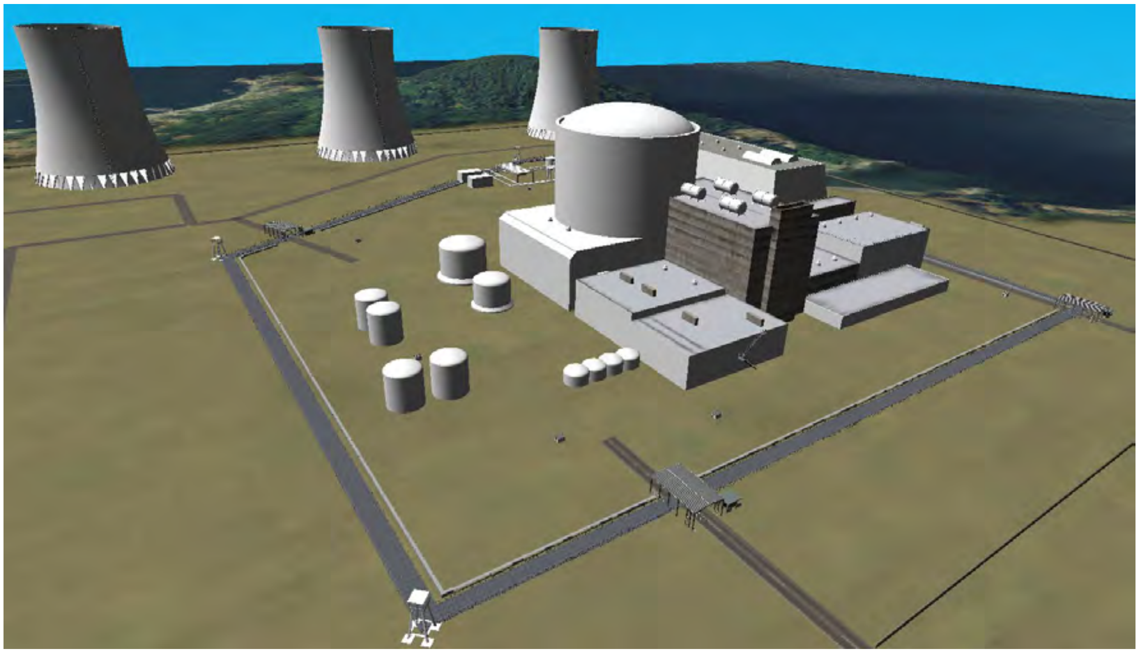


Figure 2.4 LPNPP Displayed through Presagis Creator

In 2015, as part of the International Nuclear Materials Management (INMM) vulnerability assessment tool summit, Lone Pine was chosen as the hypothetical facility for commercial and national lab entities to use for example analysis reports using a suite of software tools. Sandia National Laboratories worked closed with Rhinocorp (maker of Simajin) and ARES Security Corporation (developer of AVERT) to create models in three different tool packages: STAGE (Figure 2.5), Simajin (Figure 2.6), and AVERT (Figure 2.7).

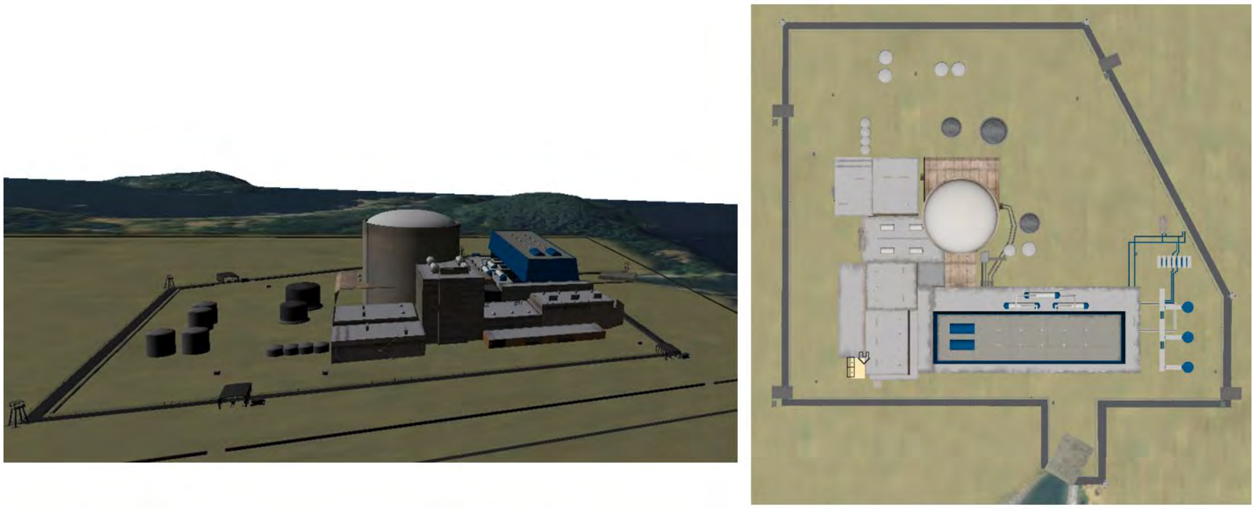


Figure 2.5 SNL LPNPP STAGE Model

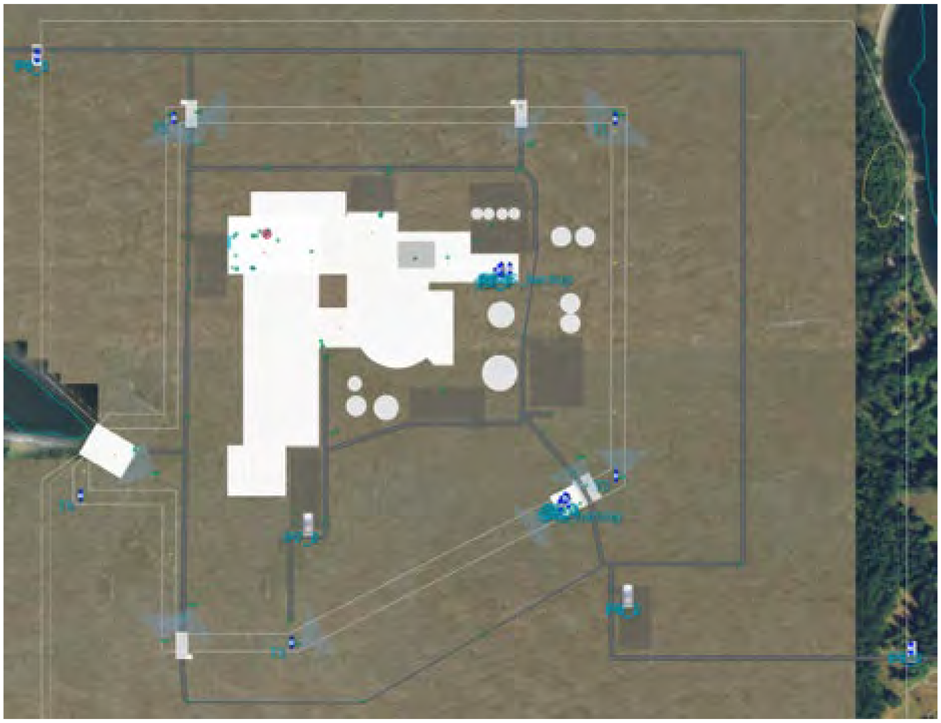


Figure 2.6 RhinoCorp LPNPP Simajin Model

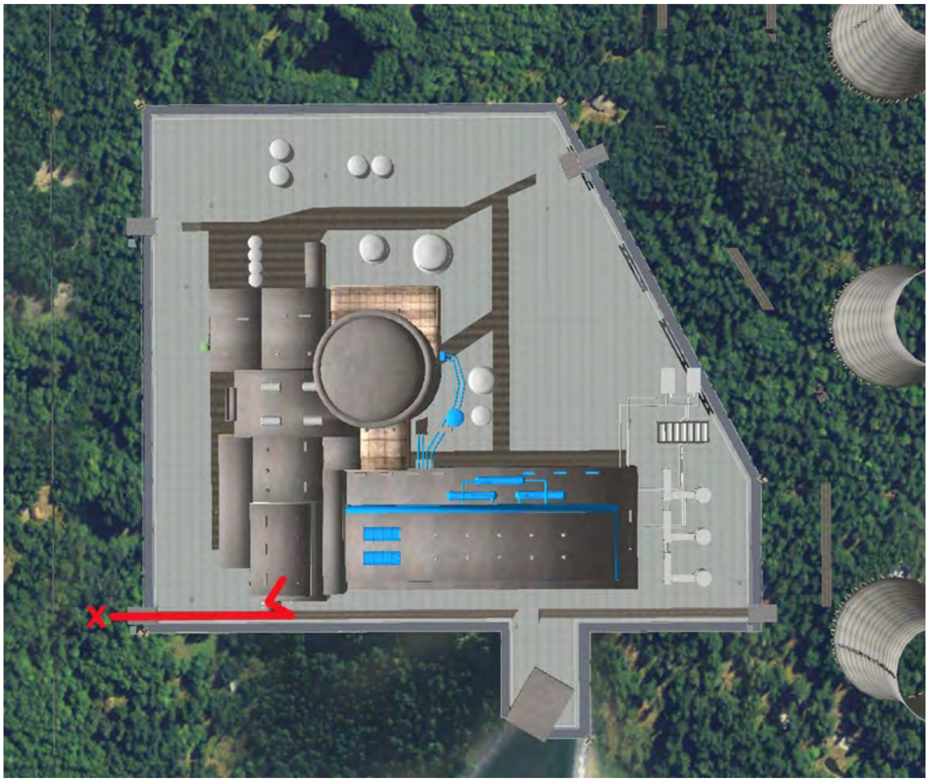


Figure 2.7 ARES LPNPP AVERT Model

In 2016, SNL incorporated the Lone Pine Facility into the 26th international training course. Since 1978, the International Training Course (ITC) has provided almost 1,000 global participants with the knowledge and practical skills to effectively analyze, design, and evaluate physical protection systems to prevent radiological sabotage and nuclear material theft. The three-week course on Physical Protection of Nuclear Facilities and Materials, prepared and delivered by Sandia National Laboratories under sponsorship of the National Nuclear Security Administration (NNSA), is now the International Atomic Energy Agency's (IAEA) flagship training course on physical security. For this 26th ITC, SNL developed a Lone Pine model and facility overview (Figure 2.8) which were featured as one of three facilities for which students designed and evaluated physical protection systems.

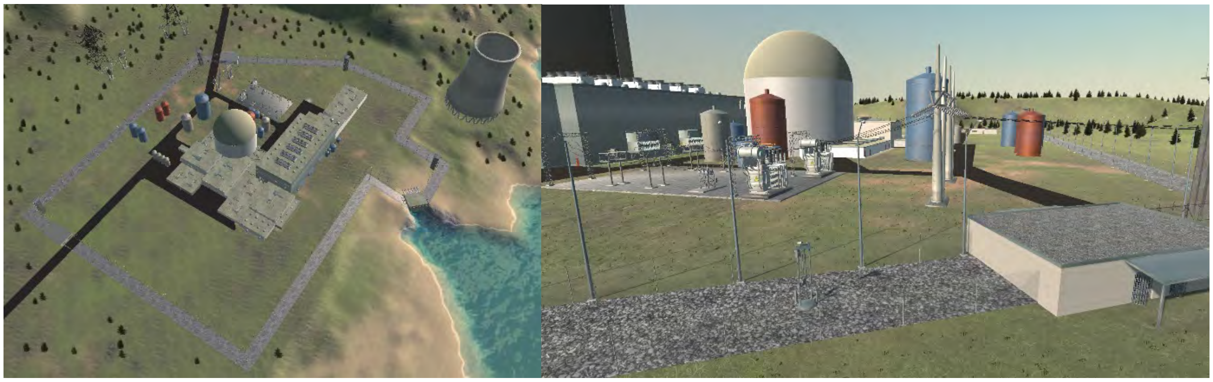


Figure 2.8 LPNPP SNL Blender Model

From 2017 through 2018, in support of the 27th ITC, SNL made visual improvements to the Lone Pine model improving its texturing. From 2018 through 2019, under the DOE-NE Light Water Reactor Sustainability (LWRS) Program, the model was improved greatly from both in terms of realism as well as visual fidelity. The subsequent sections provide these updates made to the LPNPP model funded by LWRS.

2.3 Improvements to LPNPP Reactor Coolant System

The LPNPP reactor coolant system (RCS) was initially modelled as a Westinghouse design, which lacked realism; see Figure 2.9. The RCS has since been updated for texture (Figure 2.10) and now reflect the Babcock Wilcox design of the Three Mile Island Unit 2 (TMI-2) RCS; see Figure 2.11.

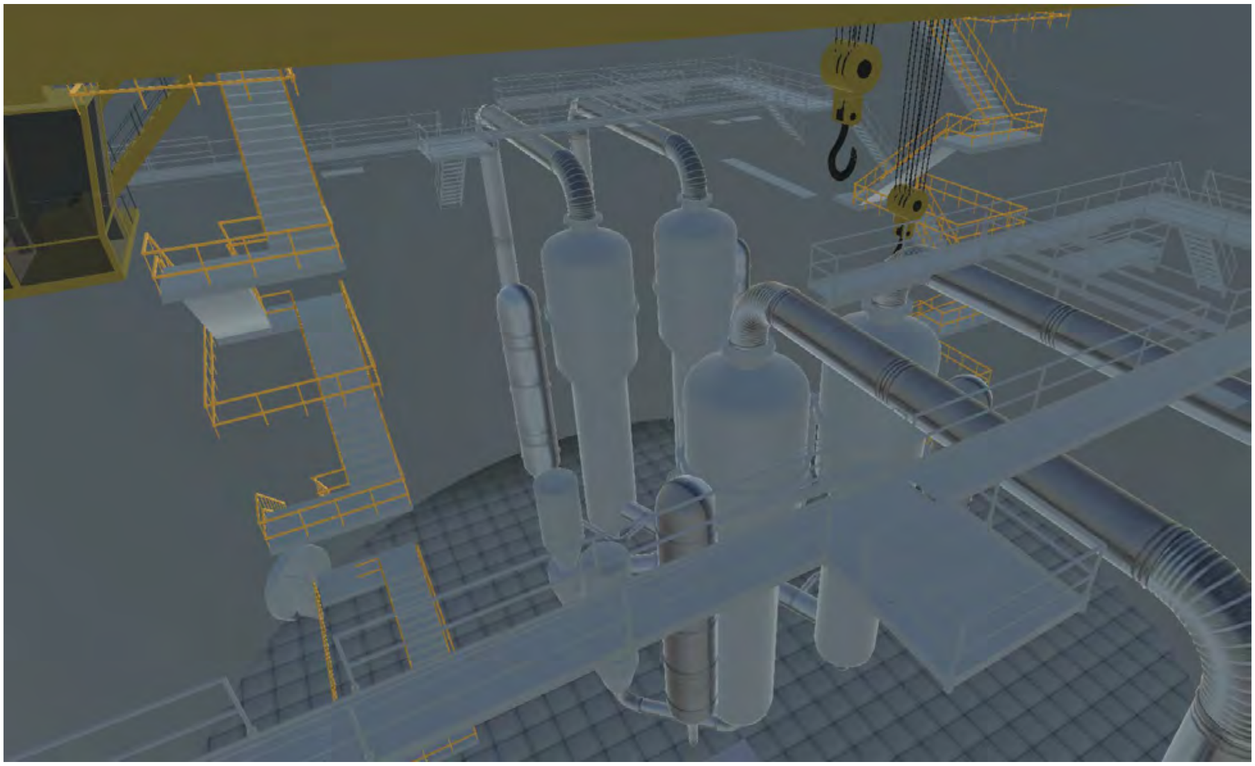


Figure 2.9 2016 LPNPP RCS Model

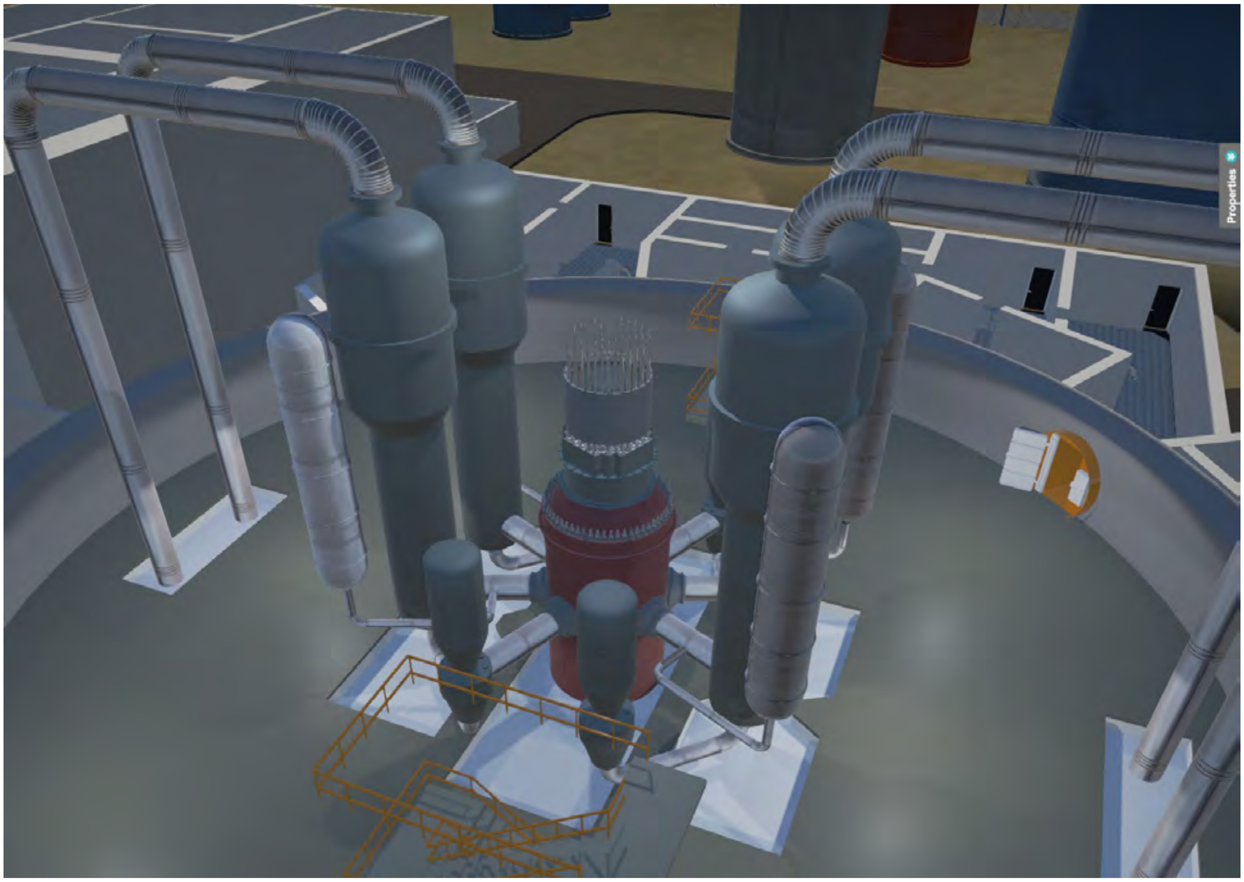


Figure 2.10 2017 LPNPP RCS Model with Improved Visualization

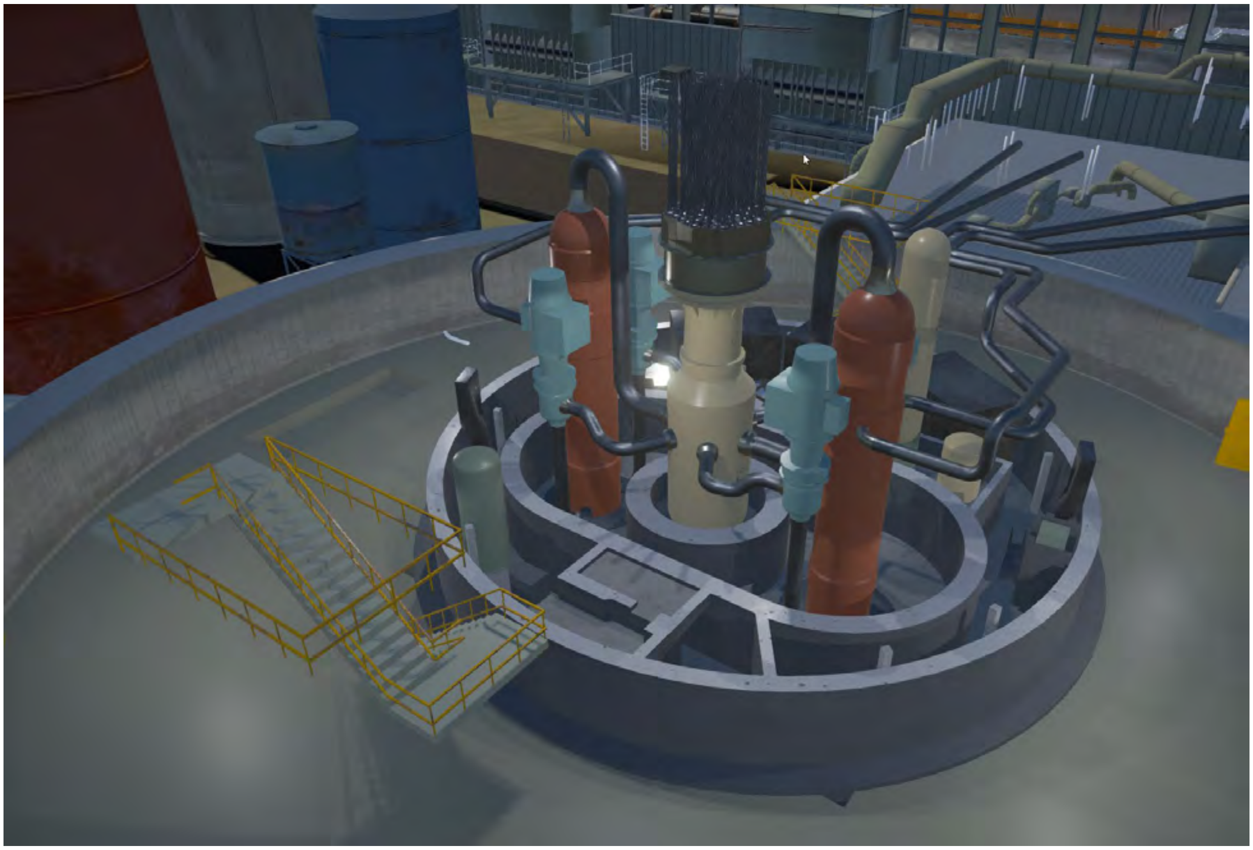


Figure 2.11 Current LPNPP RCS Model

2.4 Improvements to LPNPP Turbine Hall

Initially, the turbine hall was overly simple. In 2018, additional ducting, piping and equipment was adding the lower level to increase visual fidelity; see Figure 2.12.

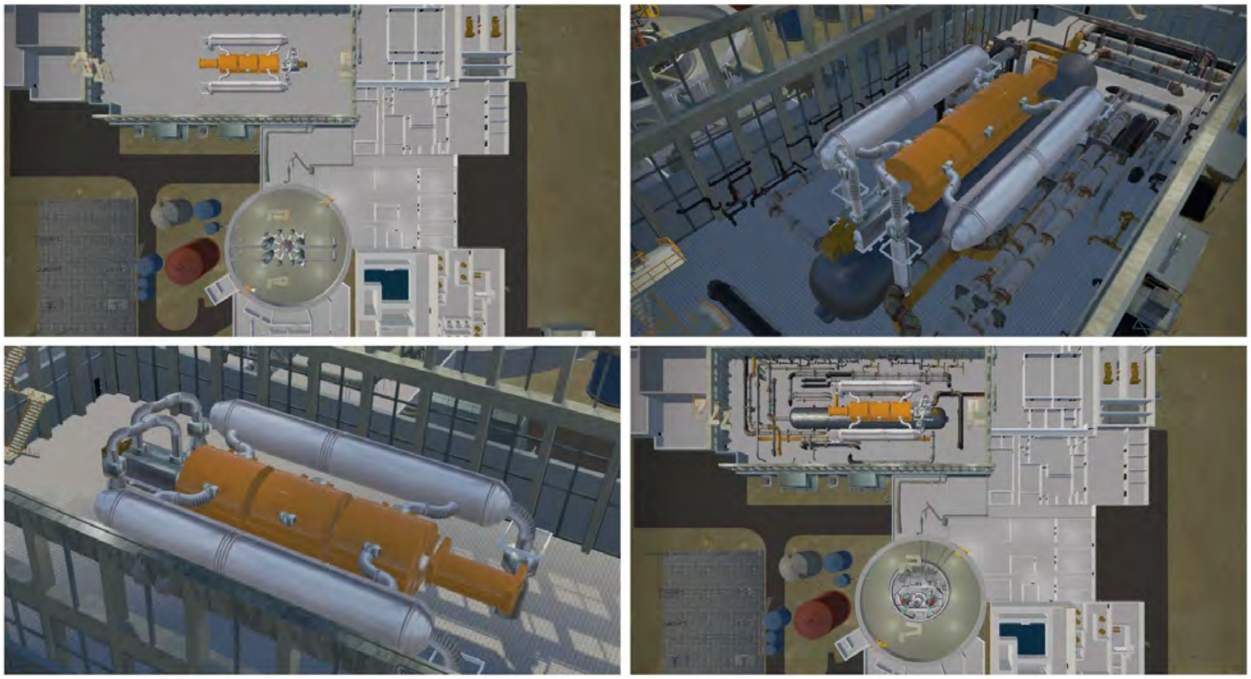


Figure 2.12 2017 LPNPP Turbine Hall (left), and Current LPNPP Turbine Hall (right)

2.5 LPNPP FLEX Building, CAS, and Guard Force Ready Room

In 2019, buildings were added to house FLEX equipment, a central alarm station (CAS), and adjacent to the CAS a guard force ready room; shown in Figure 2.13.

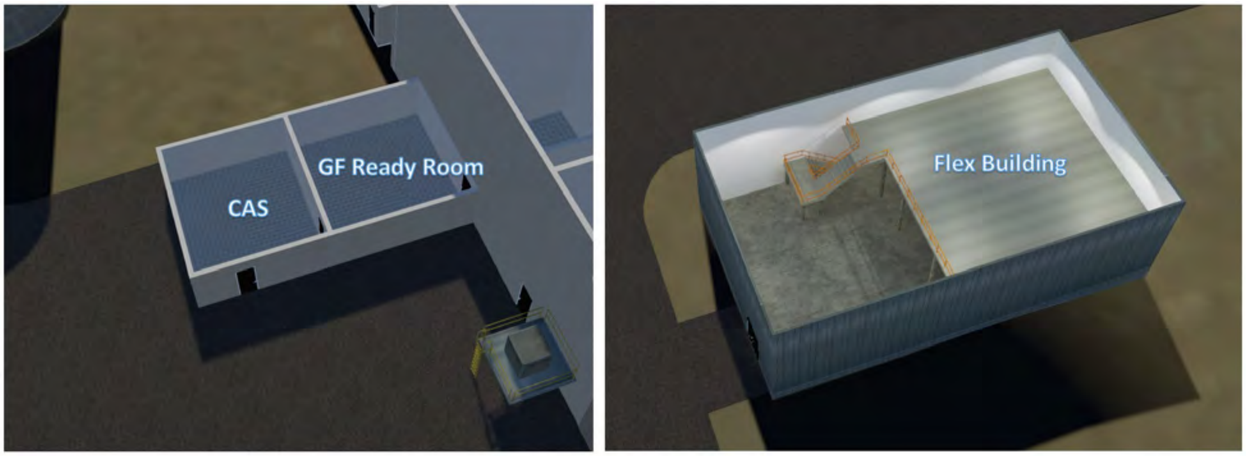


Figure 2.13 LPNPP CAS and Guard Force Ready Room (left), and FLEX Building

2.6 Improvements to LPNPP Control Room

Several updates were made to the LPNPP control room. Earlier versions of the model featured the control room on the ground floor. For the current version, it was moved to the second floor. Further control room features include a thicker wall adjacent to the turbine hall. This wall was thickened to provide shielding from any potential energetic event emanating from the turbine hall (turbine blade failure). Doors and hallways within the facility were modified such that the facility layout and access to the control room was more natural. The current version of the model features more hallways and doorways, rather than a series of connected rooms; see Figure 2.14.

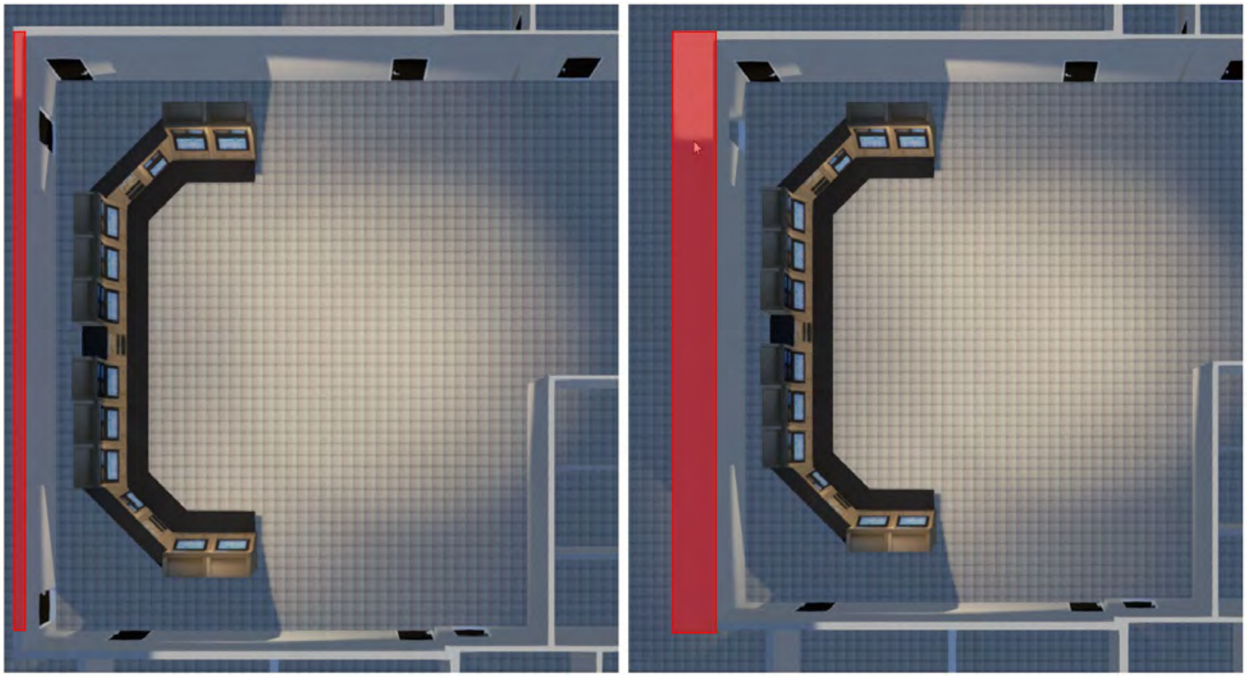


Figure 2.14 LPNPP Control Room
(wall thickness change is shown in red)

2.7 Minor Improvements to LPNPP

Additional minor updates to the LPNPP model were also conducted to include:

- Created workable model within the Scribe3D© software
- Improved overall visuals including lighting
- Modified terrain to better match external roads
- Updated doors on the interior to make navigation to the control room possible
- Updated collision geometry to better reflect facility layout
- Updated several materials to be compliant with Physically Based Render pipeline
- Updated Navigation data for better pathing around the exterior of Lone Pine and interior of the facility

2.8 Final LPNPP Model

Figure 2.15 through Figure 2.12 display the current overall LPNPP site layout. This is the current model displayed in the Scribe3D software and will be used for future LWRS physical security modeling, reactor system response modeling, and table top exercises.

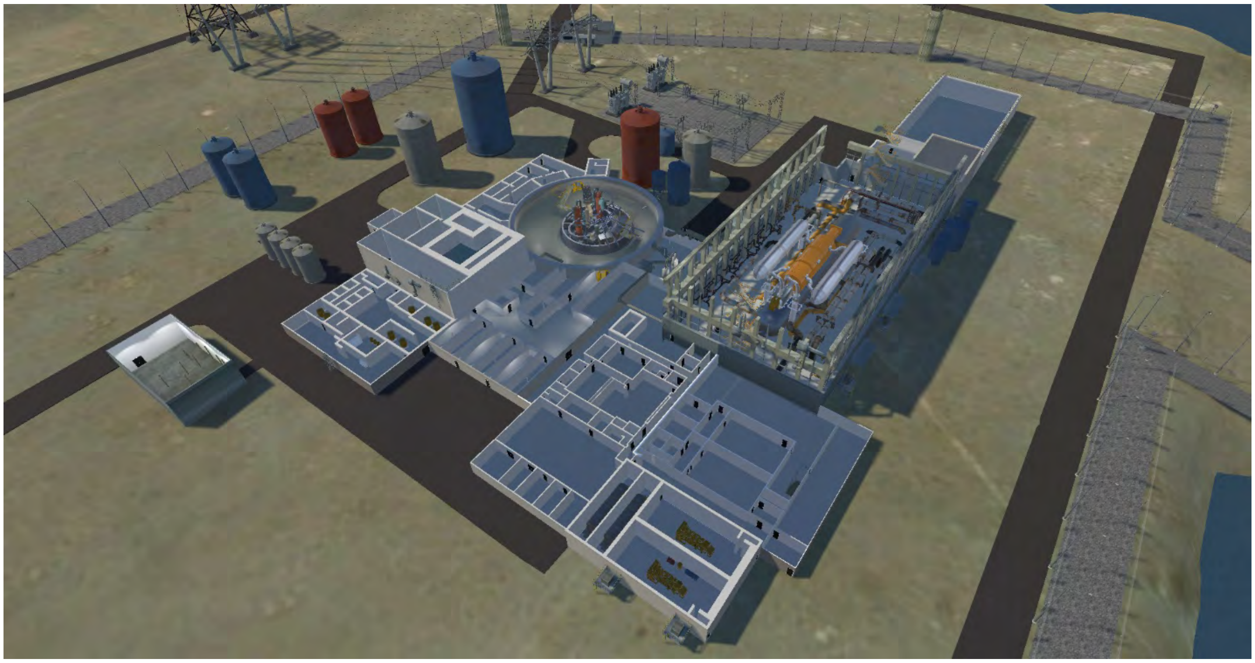


Figure 2.15 LPNPP Facility Layout

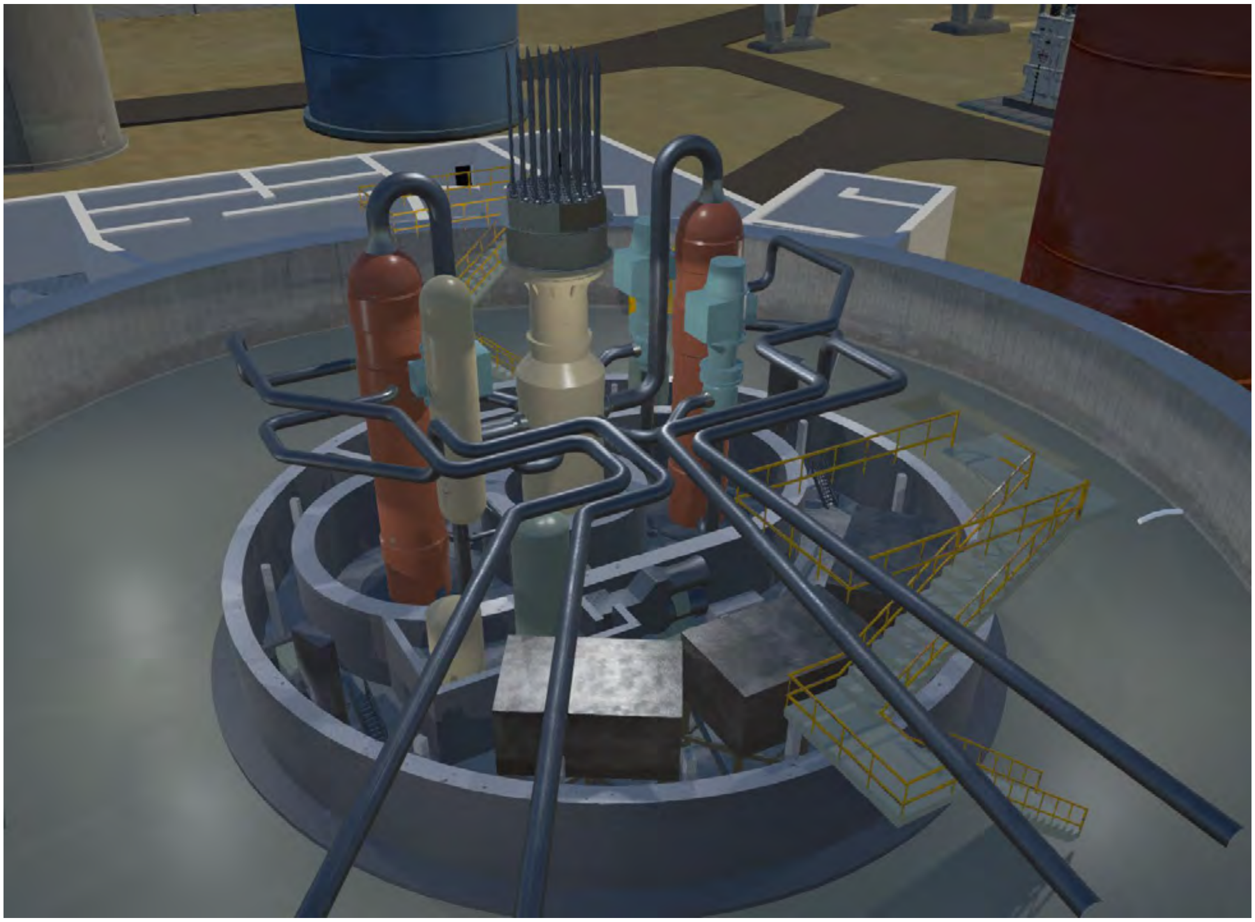


Figure 2.16 LPNPP Containment Layout

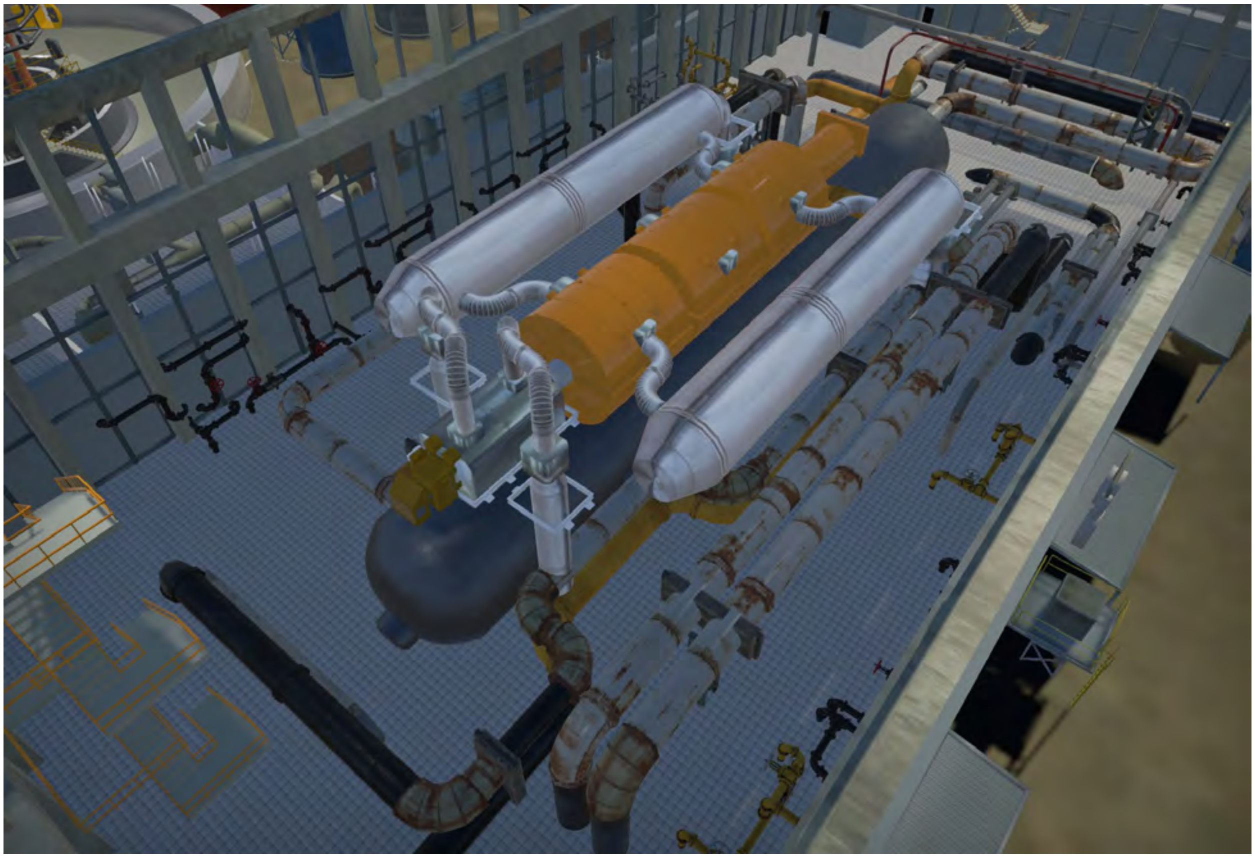


Figure 2.17 LPNPP Turbine Hall Layout

3. MELCOR

MELCOR is a fully integrated, engineering-level computer code that models the progression of severe accidents in light-water reactor nuclear power plants [8]. MELCOR is being developed at SNL for the U.S. Nuclear Regulatory Commission (NRC) as a second-generation plant risk assessment tool, and the successor to the Source Term Code package. A broad spectrum of severe accident phenomena in both BWRs and PWRs is treated in MELCOR in a unified framework. These include thermal-hydraulic response in the reactor coolant system, reactor cavity, containment, and confinement buildings; core heat-up, degradation, and relocation; core-concrete attack; hydrogen production, transport, and combustion; fission product release and transport behavior. MELCOR applications include estimation of severe accident source terms, and their sensitivities and uncertainties in a variety of applications. Design basis accidents in advanced plant designs (e.g., the Westinghouse AP-1000 design and the GE Hitachi Nuclear Energy ABWR design) have been analyzed with MELCOR.

Current applications of MELCOR include the NRC sponsored State-of-the-Art Reactor Consequence Analyses (SOARCA) [5][7], and the U.S. Department of Energy (DOE) sponsored Fukushima Daiichi accident analyses [6].

3.1 Overview

MELCOR can estimate the fission product source term. MELCOR can also apply sensitivity and uncertainty analysis for the estimated source term. Additionally, MELCOR is divided into 20 different packages and an execution primer. All of these packages are coupled within the code to model major reactor plant systems. The codes response to accident conditions include but are not limited to [8]:

- Thermal-hydraulic response of the primary reactor coolant system, the reactor cavity, the containment, and the confinement buildings,

- Core uncovering, fuel heatup, cladding oxidation, fuel degradation, and core material melting and relocation,

- Heatup of reactor vessel lower head from relocated fuel materials and the thermal and mechanical loading and failure of the vessel lower head and transfer of core materials to the reactor vessel cavity,

- Core-concrete attack and ensuing aerosol generation,

- In-vessel and ex-vessel hydrogen production, transport, and combustion,

- Fission product release, transport, and deposition,

- Behavior of radioactive aerosols in the reactor containment building,

- Impact of engineered safety features on thermal-hydraulic and radionuclide behavior.

The MELCOR code uses a ‘control volume’ approach for describing and combining reactor plant systems. There are no specific ‘nodes’ that a user must incorporate, and thus allows for a greater degree of freedom. With this in mind, it is possible for MELCOR to provide a detailed and unique reactor plant model for any type of pressurized water reactor or boiling water reactor and has even been proven to successfully model Russian VVER and RMBK-reactor classes [8].

The first part of the MELCOR execution is called MELGEN. MELGEN provides a starting point for MELCOR. The majority of the initial conditions are specified, processed, and checked for execution errors. Upon execution of MELGEN, a restart file for MELCOR is written. The MELCOR code is then executed using this restart file and advances the accident scenario through predetermined time steps until a prespecified end time is achieved. As part of the MELCOR/MELGEN output, a plot file (.PTF) is created. The MELCOR output variables are written to this plot file at predetermined time intervals set by the user. The plot file can be read using an Excel macro program, or it can be converted into a text file which can be read by the post processing analysis or ADAPT.

3.2 Station Blackout Scenario

For this effort, a station blackout scenario with and without reactor protection systems (e.g., station batteries for DC power and turbine driven auxiliary feedwater) was considered. The worse-case scenario for sabotage is a station blackout without reactor protection systems. This scenario is further discussed.

Upon loss of offsite AC power without reactor protection systems, the turbine, the reactor coolant pumps, the makeup pumps, and the main feedwater pumps shut down. The diesel generators attempt to auto-start to provide emergency AC power but will fail; this is a loss of onsite AC power and yields a station blackout. With no AC power, the reactor coolant pumps coast down to a stop, the flow rate of coolant through the core decreases. This causes the temperature difference between the primary and secondary circuits to increase initially increasing heat transfer between the primary and secondary circuits.

The rate of steam production in the steam generators increases and the steam line pressure increases, actuating the atmospheric steam dump valves. However, the auxiliary feedwater system does not provide any volume of feedwater to maintain steam generator level at nominal reactor power. With no feedwater, the level in the steam generators drops causing the heat transfer between the primary and secondary system to decrease.

The primary coolant temperature and pressure continue to rise; until eventually, the fuel temperature and cladding temperature design limits are exceeded. Analyses of this postulated accident are discussed further in Section 3.4.

3.3 MELCOR Model of LPNPP

The Three Mile Island Unit 2 (TMI-2) MELCOR model was originally created for code verification.[a] The model has since undergone revisions to reflect state-of-the-art PWR severe accident best practices. For this work, the TMI-2 sequence boundary conditions were replaced with generic short-term station blackout (STSBO) scenario within the LPNPP. This work presents a novel analysis of a total loss of alternating current (AC) and direct current (DC) power at the LPNPP using the MELCOR severe accident analysis code. Additional information on the LPNPP reactor and steam plant systems can be found in Reference [3].

3.3.1 Reactor Core and Pressure Vessel

Figure 3.1 shows the LPNPP reactor pressure vessel (RPV) nodalization, and Table 3.1 lists design parameters for the reactor core and RPV. In Figure 3.1, “CV” denotes hydrodynamic control volumes, and “COR” represents individual cells that contain reactor core components (i.e. fuel assemblies, control rods, supporting steel).

Table 3.1 Design Parameters for LPNPP

Parameter	Value
Rated Core Power	2771.97 MWth
UO2 Fuel Mass	95383.0 kg
Zircaloy Cladding Mass	23026.0 kg
RPV Inner Diameter	2.172 m
RPV Height	11.374 m

The LPNPP reactor core is represented by five concentric rings with each ring divided into four vertically stacked hydrodynamic control volumes. Each control volume contains three COR cells. The axial length of the fuel assemblies is evenly divided between 12 axial cells per ring. Core component mass is radially apportioned based on the number of fuel assemblies in each ring. The component mass in each ring is evenly distributed across the axial cells. The outermost ring (i.e. Ring 5) models the core shroud and bypass region.

The RPV model uses a detailed hydrodynamic nodalization to represent the lower and upper plenums. The lower plenum consists of two stacked control volumes. Both control volumes contain five COR cells that model the core support plate, flow mixers, and core support columns. The upper plenum is axially divided into multiple control volumes that represent the control rod guide tubes and upper head.

[a] On March 28, 1979, the Three Mile Island Unit 2 nuclear power plant experienced a partial core meltdown and is the most serious accident within the United States commercial nuclear power plant operating history.

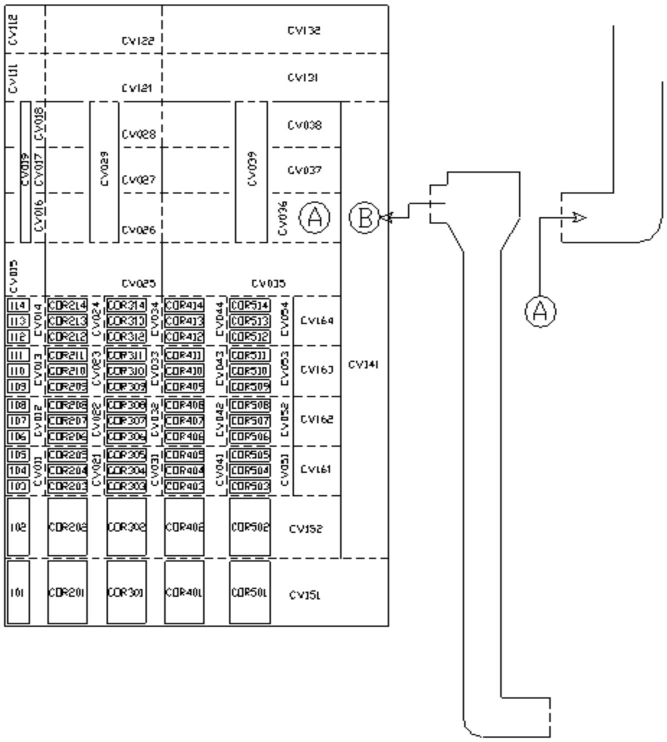


Figure 3.1 LPNPP RPV Nodalization

The LPNPP model uses the default MELCOR radionuclide (RN) inventories and are shown with representative RNs for each class in Table 3.2. A full list of radioisotopes within each MELCOR class is listed in Table 3.3. The shutdown decay heat is based on these radionuclide masses and is calculated using the American Nuclear Society decay heat standard.

Table 3.2 Radionuclide Reactor Core Masses for LPNPP

MELCOR Class	MELCOR RN Class	Mass (kg)
1	Xe	2.779E+02
2	Cs	1.549E+02
3	Ba	1.219E+02
4	I_2	1.197E+01
5	Te	2.439E+01
6	Ru	1.716E+02
7	Mo	2.023E+02
8	Ce	3.570E+02
9	La	3.312E+02
10	UO_2	8.242E+04
11	Cd	8.101E-01
12	Ag	4.601E+00
13	CsI	9.998E-07

Table 3.3 Radionuclides used for LPNPP Decay Heat

MELCOR Class	MELCOR RN Class	Radioisotopes
1	Xe	Xe, Kr
2	Cs	Cs, Rb
3	Ba	Ba, Sr
4	I_2	I
5	Te	Te
6	Ru	Ru, Rh
7	Mo	Mo, Nb, Co, Tc
8	Ce	Ce, Np, Pu, Zr
9	La	La, Cm, Am, Pr, Y, Nd
10	UO_2	UO_2
11	Cd	Cd
12	Ag	Ag
13	CsI	See Class 2 & 4

3.3.2 Reactor Coolant System

The LPNPP reactor coolant system (RCS) has two loops (Loop A and Loop B) with their own steam generator (SGs) and each loop containing two reactor coolant pumps (RCPs). Figure 3.2 illustrates the LPNPP MELCOR model hydrodynamic nodalization of the RCS; including the RPV, SGs, RCPs, and the pressurizer. For a single loop, coolant flows out of the RPV, through a hot leg, and to the steam generator where it is used to produce steam for the steam turbine power conversion system. The coolant then flows out of the steam generator and to the cold leg where it is pumped back to the RPV by the RCPs.

The SG model includes representations of the SG tubes, the upper and lower plenums, the SG secondary side, and the steam lines to the steam turbine power conversion system. The primary and secondary SG sides are both represented by five stacked control volumes. Heat structures represent the SG tube bundle and allow heat transfer between the primary and secondary control volumes. Two control volumes represent the upper and lower plenums, respectively.

The pressurizer is connected to Loop A. Like the SGs, the pressurizer also has a five axial control volume representation. A single flow path connected to the uppermost volume provides a lumped representation of the primary safety valves (SVs). During severe accident conditions, the SVs vent to a pressure relief tank (PRT) within the containment.

During an STSBO, the accumulators and the turbine drive auxiliary feedwater system (TDAFW) and the only safety systems available. The accumulators discharge into the Loop A hot leg, and the TDAFW provides coolant to the secondary side of both SGs. Both safety systems were mechanistically modeled as mass sources within their respective control volumes.

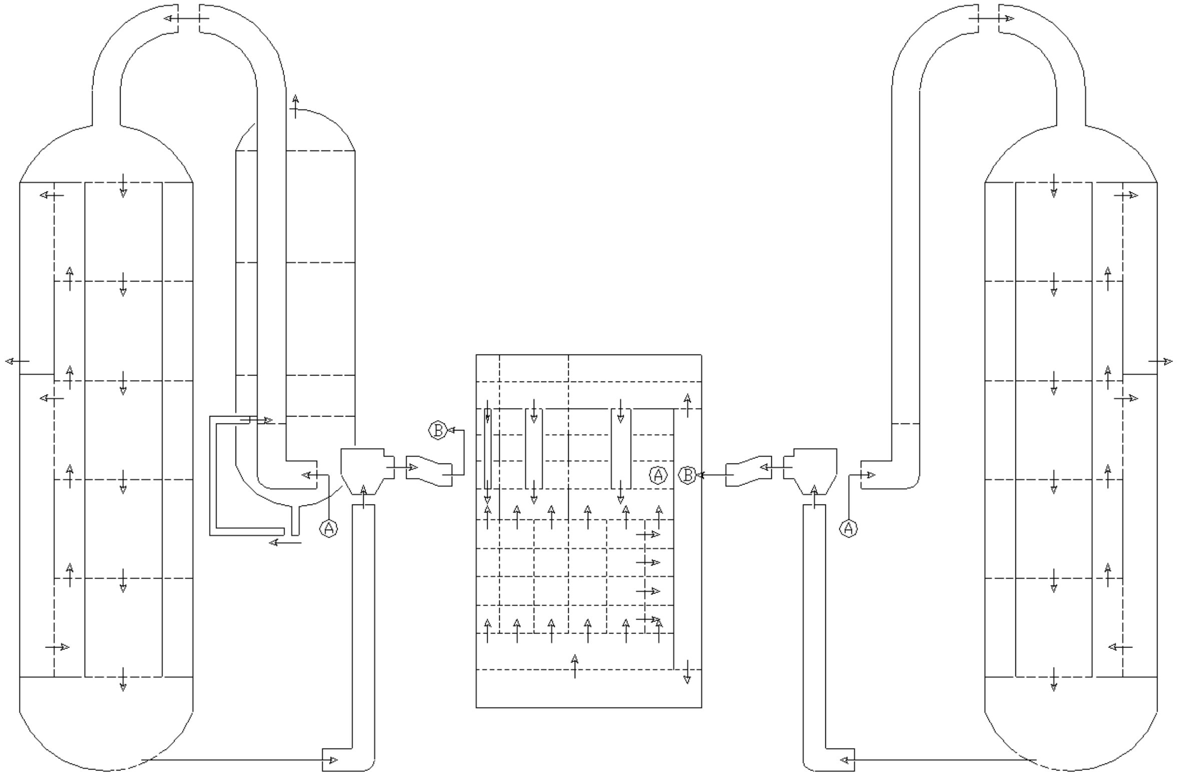


Figure 3.2 LPNPP RCS Nodalization

3.3.3 Reactor Building and Containment

The LPNPP model uses a simplified containment and reactor building representation. The containment has four CVs. Of the four containment CVs, two CVs represent the SG compartments, one CV models the pressure relief tank, and the remaining CV models the reactor cavity. The reactor building is a single CV. The reactor building has heat structures representing floors, walls, ceilings, and miscellaneous structures. A single flow path models nominal leakage out of the reactor building CV.

The PRT is located inside the containment. If the PRT over pressurizes, a burst disk will open a flow path between the PRT and the containment. This allows direct transport of radioactive material from the primary system to the containment prior to failure of the RCS or RPV.

3.4 MELCOR Short-Term Station Blackout of LPNPP

Table 3.4 summarizes the event timings in the LPNPP STSBO. The accident sequence begins with the loss of all onsite and offsite AC power and a loss of DC power (see Section 3.2). The reactor successfully trips (Scram) and isolates appropriate at-power support systems, but all powered safety systems are unavailable. The accident progresses quickly with the first onset of fission product release at ~30 minutes and lower head failure (LHF) at 1.11 hours. Following lower head failure, the plant reaches "steady-state" severe accident conditions, and no significant events occur. Section 3.4.1 and Section 3.4.2 discuss the reactor plant response and radionuclide release, respectively, in more detail.

Table 3.4 STSBO Event Sequence

Event	Time (hours)
Loss of all onsite and offsite AC and DC power	0.0
Reactor scram	0.0
PRT rupture disk opens	0.14
RPV level is below the top of active fuel	0.23
First fission product release	0.51
Onset of fuel damage	0.93
Accumulators begin injection	N/A
Lower head failure	1.11
SG-A TDAFW starts	1.96
SG-B TDAFW starts	2.63
SG-A and SG-B dryout	N/A
End of simulation	24.00

3.4.1 LPNPP Thermal-Hydraulic Response

The pressure response of the primary and secondary systems is shown in Figure 3.3. Following the reactor scram, the reactor coolant pumps and the main feedwater pumps trip within seconds afterwards and coast down. Without any decay heat removal capabilities, the RPV pressure increases to the SV setpoints. The SVs maintain pressure in the RPV until the PRT rupture disk bursts at 0.14 hours. The elevated RPV pressure, due to the thermal surge of decay heat within the RPV, does not allow the SVs to reclose until 0.62 hours. After 0.62 hours, the primary SVs begin to cycle again until lower head failure of the RPV depressurizes the RCS.

In contrast to the RPV, the SG pressures are relatively smooth until TDAFW injection at ~2 hours (Figure 3.3). Initially, the coolant in the SGs boils off and increases the SG pressure. However, after the coolant boils off, the SG pressures steadily decrease until lower head failure of the RPV. It is unclear what causes the SGs to depressurize following LHF as the primary and secondary systems are not thermo-hydraulically coupled; this will be investigated in future work.

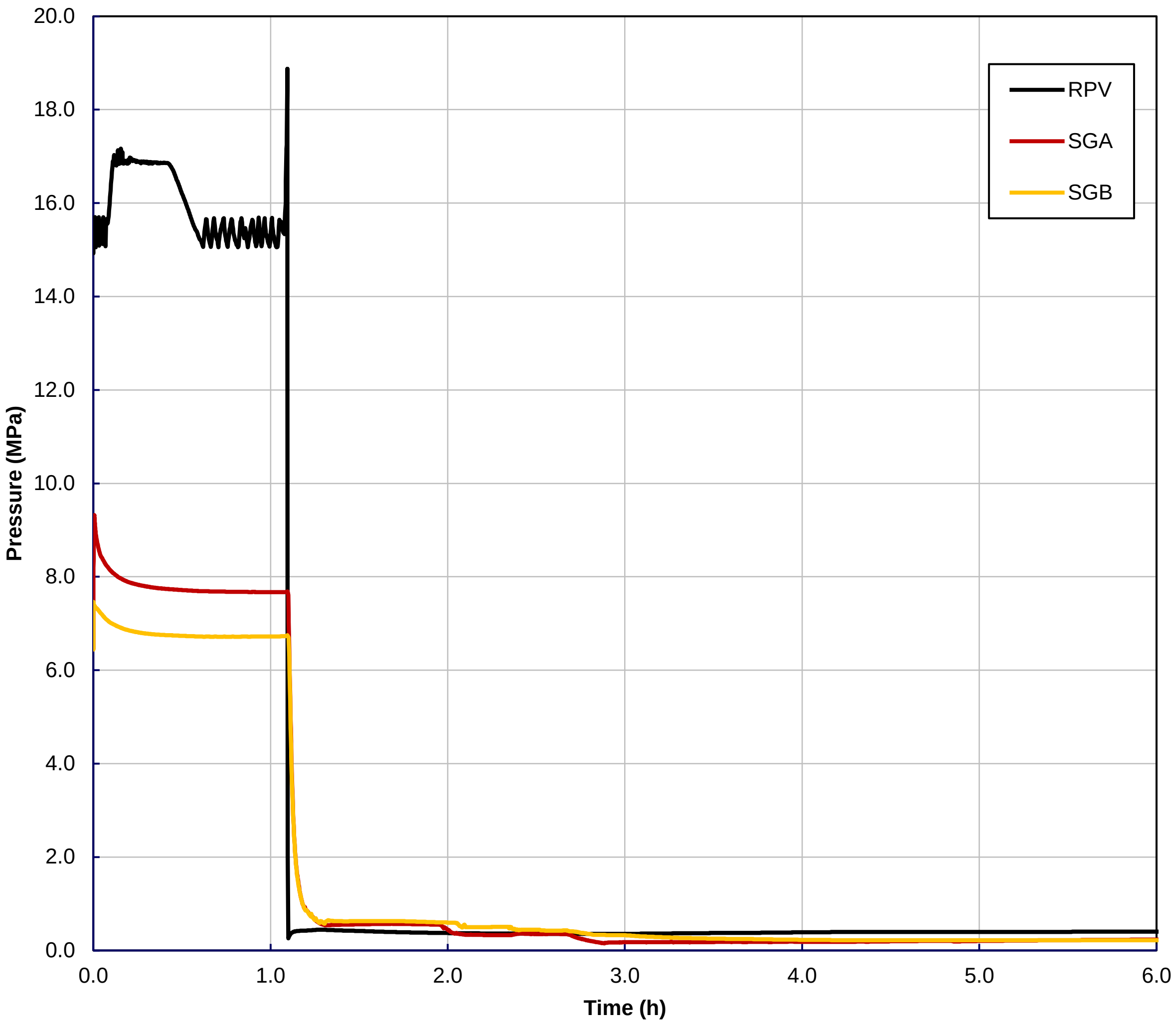


Figure 3.3 Primary and Secondary Pressure Response

The RPV and pressurizer water levels are shown in Figure 3.4. After the reactor scram, the primary coolant heats up, expands, and swells to the top of the pressurizer. The coolant flowing through the pressurizer SVs causes over-pressurization of the PRT and failure of the PRT rupture disk at 0.14 hours. With the SVs unable to reclose after PRT rupture disk failure, coolant freely flows out of the primary system, and the RPV level subsequently decreases (Figure 3.4). At 0.21 hours, the core begins to uncover, at 0.51 hours the onset of core damage begins, and by 0.69 hours the RPV is below the bottom of the fuel. The RPV level swells up at 1.09 hours from hot core debris falling into the lower plenum and lower head failure of the RPV occurs at 1.10 hours. The RPV completely dries out at 1.60 hours.

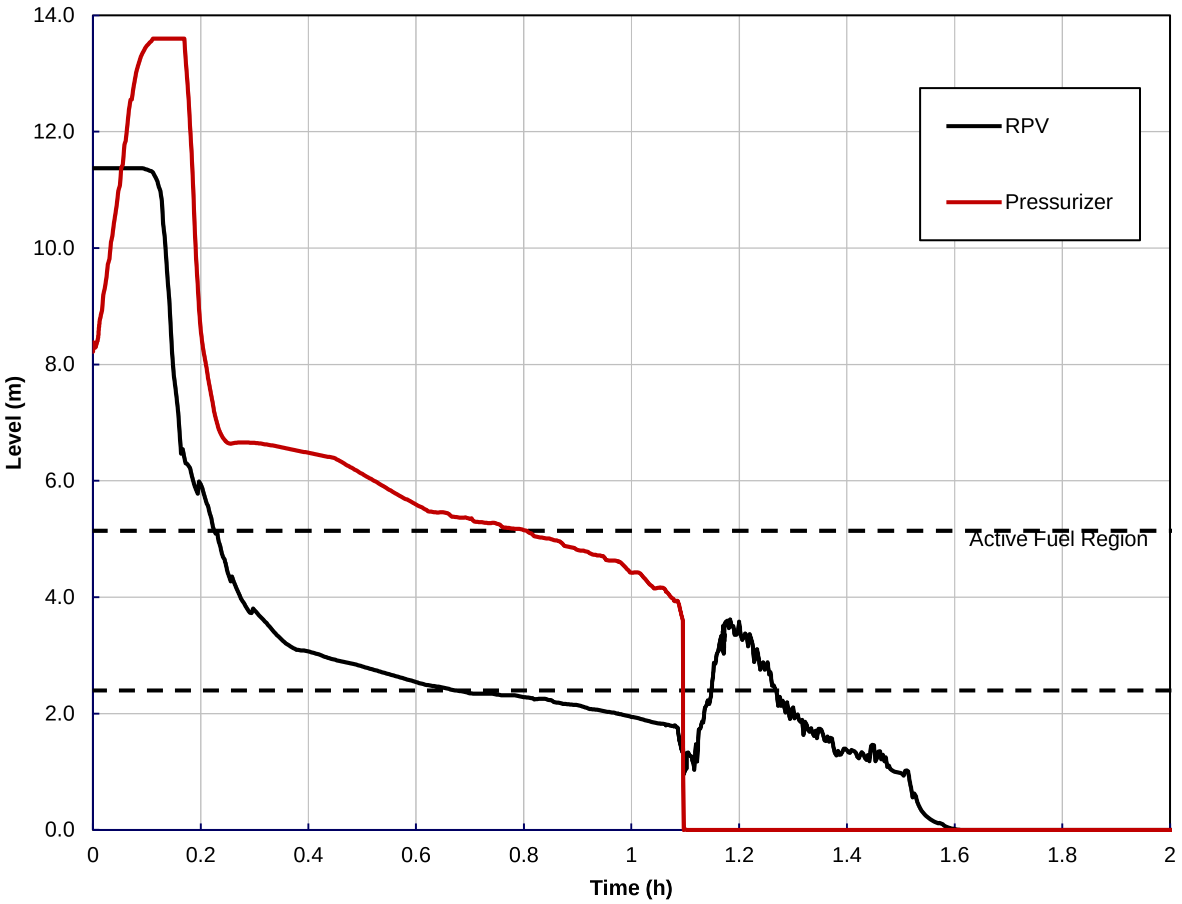


Figure 3.4 RPV and Pressurizer Coolant Levels

Figure 3.5 shows the SG water levels. Unlike the primary system, the SG water levels maintain at relatively constant coolant level until a brief transient at 1.09 hours. It is unclear what causes this transient, although it is likely related to hot steam produced by core degradation flowing through the SG tubes. Since the severe accident sequences within the RPV and RCS occur as such a rapid pace, the ability to establish natural circulation does not occur; thus, the steam generator water levels remain high. Both SG levels quickly increase following their respective TDAFW actuation (see Table 3.4). It is unknown as to why SG-A has earlier TDAFW injection than SG-B; this will be further investigated in future work.

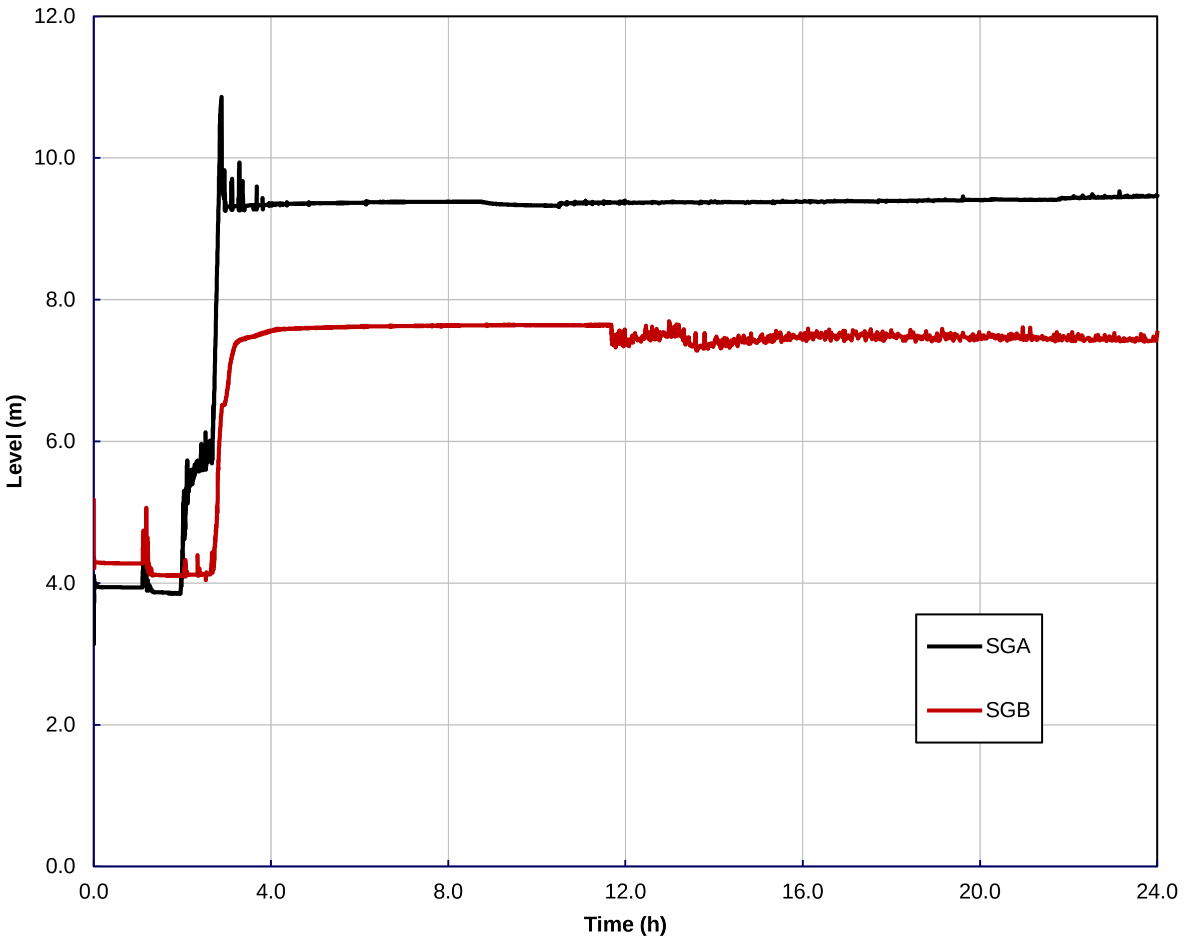


Figure 3.5 Steam Generator Water Levels

The peak fuel temperature is shown in Figure 3.6. The peak temperature sharply rises at 0.21 hours when the RPV water level drops below the top of the active fuel region. Fuel rod degradation (core collapse) begins once the fuel reaches 2700 K at 0.93 hours. The peak temperature decreases as the hottest fuel elements collapse into the lower plenum of the RPV at ~2 hours. After 2 hours, the remainder of the simulation shows the fuel in the peripheral of the core as it overheats but does not fully collapse.

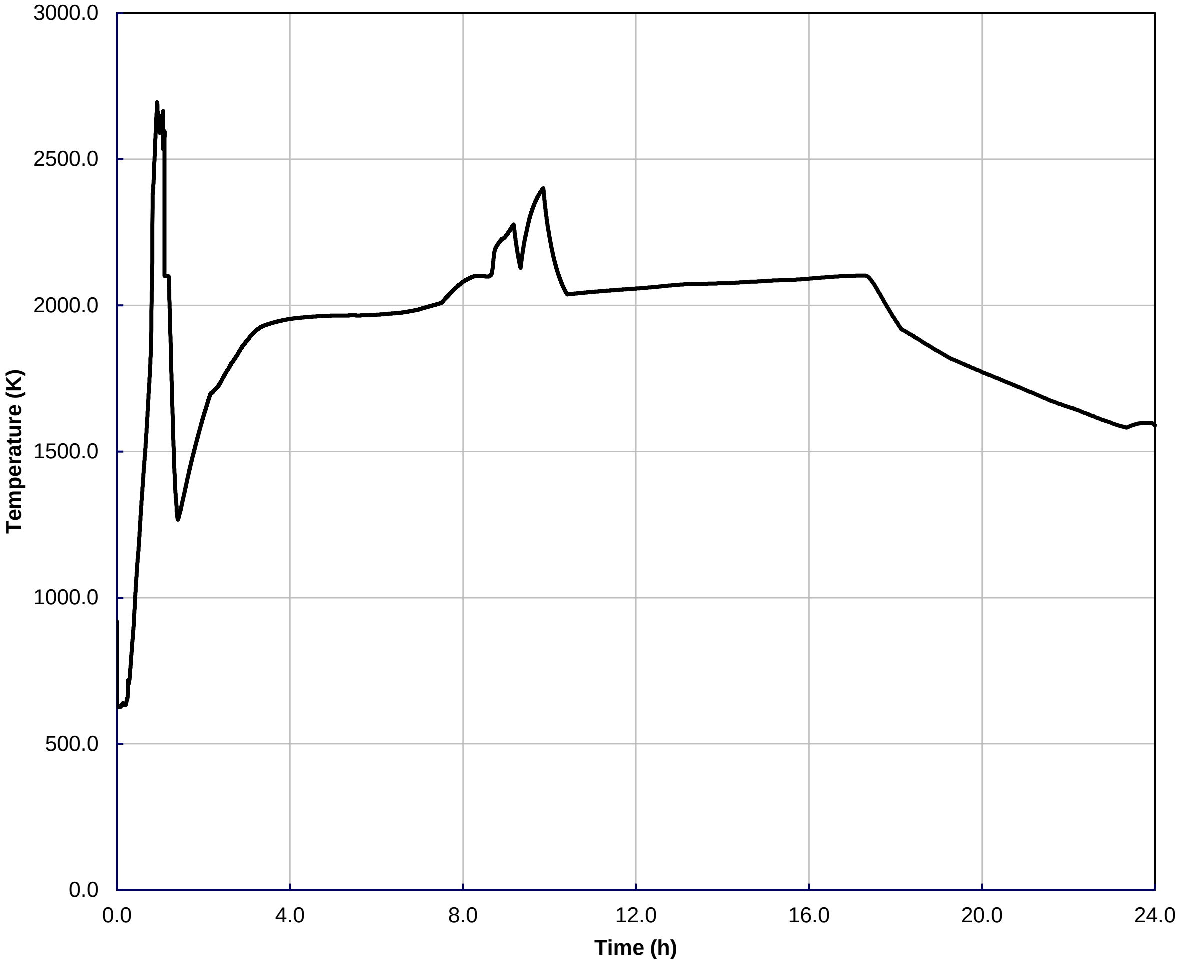


Figure 3.6 Reactor Core Peak Fuel Temperature Response

3.4.2 LPNPP Radionuclide Release

Containment leakage is the primary mechanism for radionuclide release to the environment. Figure 3.7 shows the containment pressure and the mass flow rate through the containment leakage flow path. Intuitively, the containment leakage is proportional to the containment pressure. Prior to 0.14 hours, the containment is at atmospheric pressure. After the PRT rupture disk opens, the containment pressure sharply increases as primary coolant blows down into containment. Core debris ejection from the lower head failure of the RPV and subsequent depressurization of the RCS spikes the containment pressure at 1.11 hours. Molten core debris interactions with the containment concrete in the cavity steadily increase the pressure throughout the remainder of the simulation.

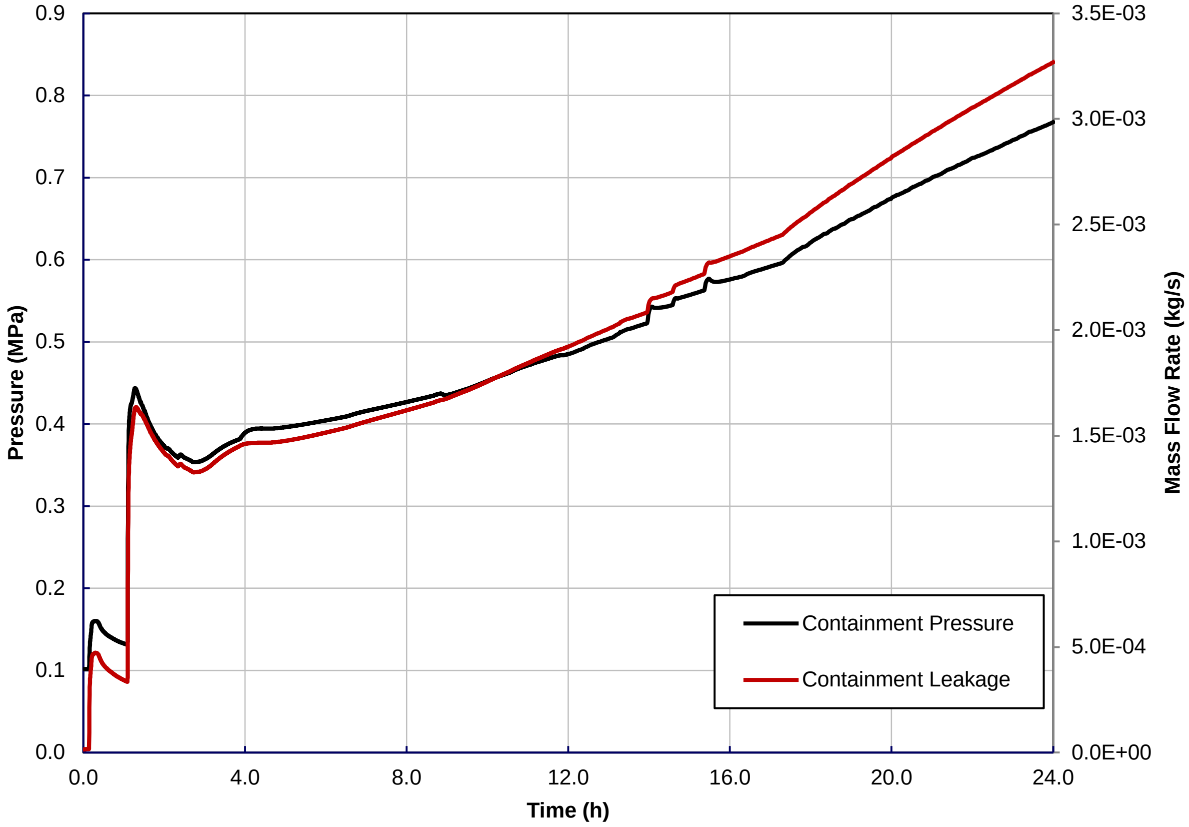


Figure 3.7 Containment Pressure Response and Containment Leak Rate

Figure 3.8 and Figure 3.9 show bar plots for the environmental release of MELCOR RN mass and fraction of core RNs, respectively, for each MELCOR radionuclide class (see Table 3.2 and Table 3.3 for additional information). Note that the Cs, CsI, I_2 classes are stoichiometrically combined to give the total Cs (i.e. ‘All Cs’) and I (i.e. ‘All I’) masses and release fractions. Compared to other PWR simulations [5,7], the total radionuclide releases are high. Notably, 13.4% of Cs (13.4 kg) and 4.6% of I (1.12 kg) release to the environment. For comparison, modern SNL PWR STSBO have yet to estimate Cs releases above 8.0% (e.g., see Reference [7]).

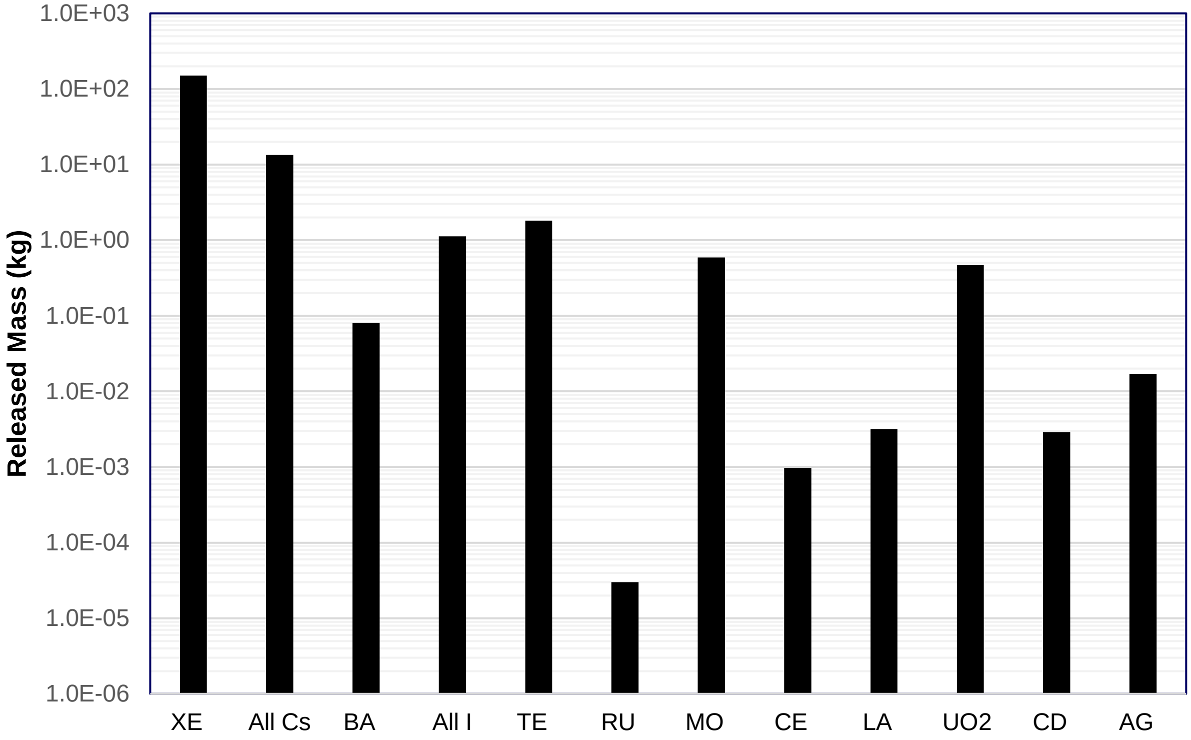


Figure 3.8 Mass of MELCOR Radionuclide Classes Released to the Environment

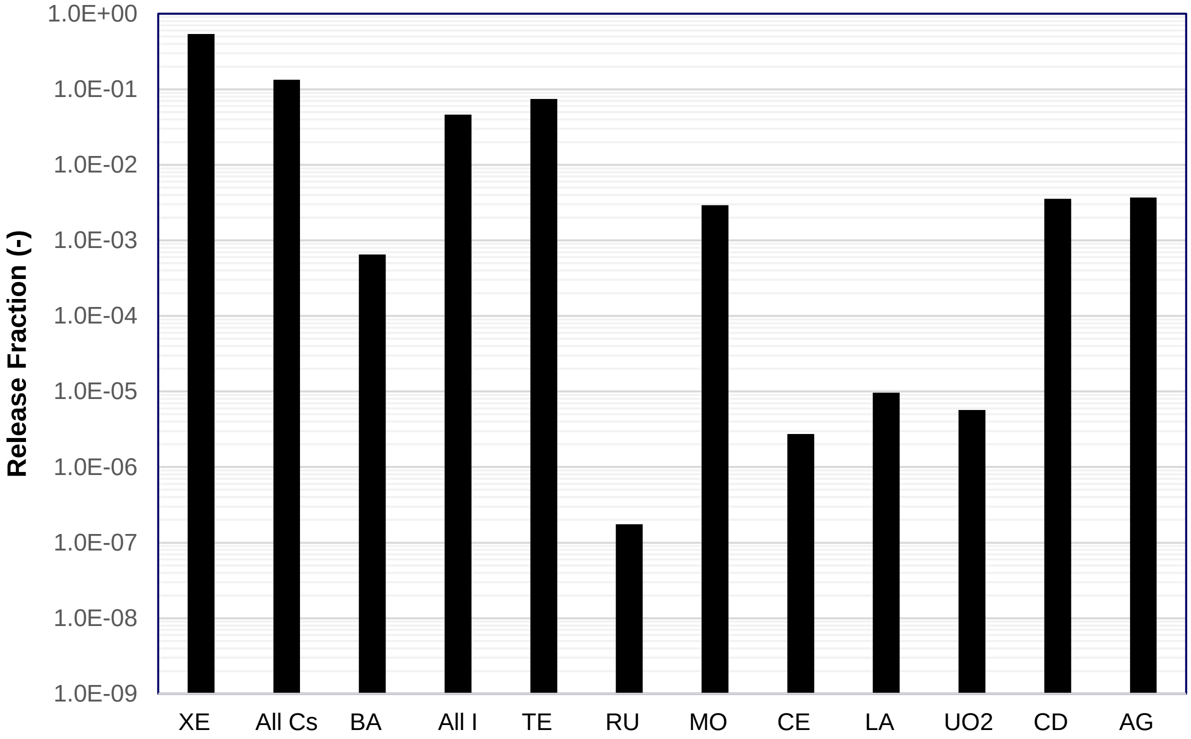


Figure 3.9 Fraction of Core Mass of MELCOR Radionuclide Classes Released to the Environment

Cs and I MELCOR classes typically dominate health consequences from a severe accident. Figure 3.10 shows the Cs and I mass released to the environment throughout the simulation. Almost immediately after the first fission product release at 0.51 hours, both Cs and I transport to the environment; as CsI vapor. Cs and I continue to release until RPV lower head failure at 1.1 hours. The released masses plateau after RPV lower head failure indicates that the remainder of the fission products settle within the RPV, RCS, and containment. This plateau represented ~4.6% of the core inventory of iodine and ~13.4% of the core inventory of cesium released to the environment.

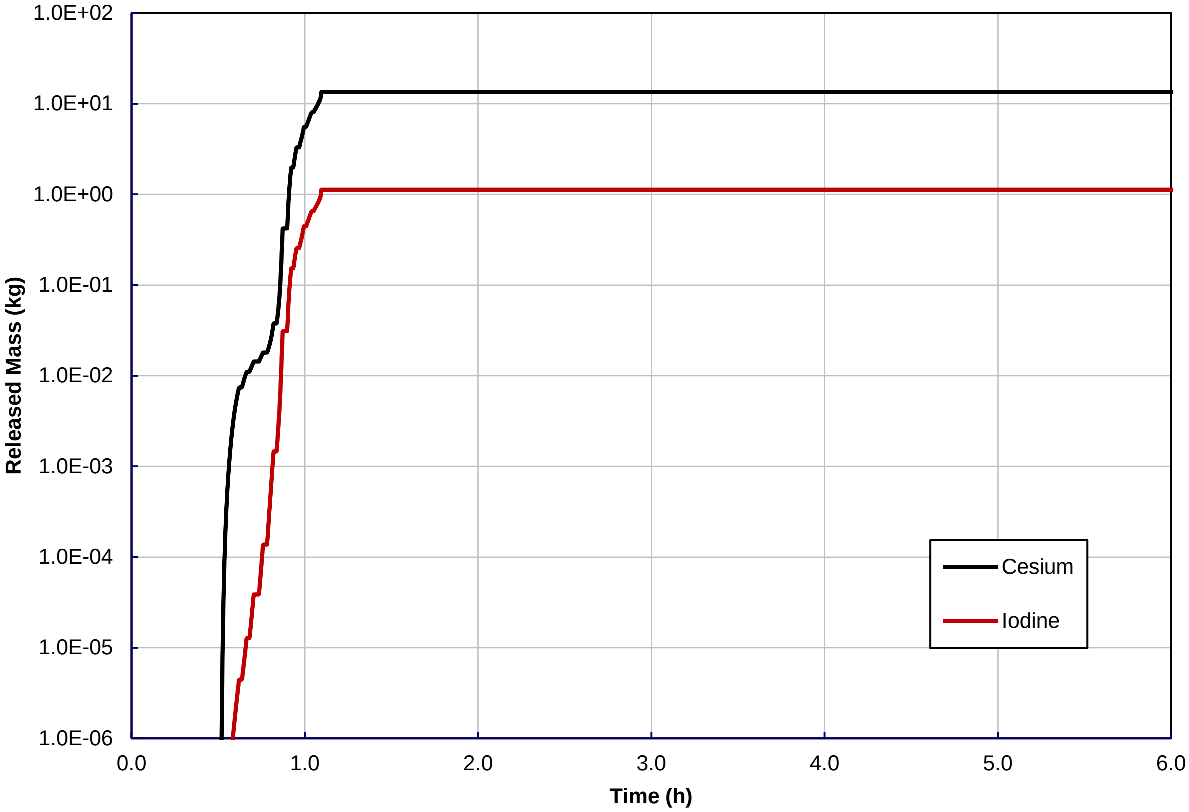


Figure 3.10 Mass of Cesium and Iodine MELCOR Radionuclide Class Response as Released to the Environment

3.4.3 STSBO Conclusions

The STSBO MELCOR simulation predicted a quick core degradation transient with core damage occurring within an hour of the initiating event. The environmental release fractions for the scenario were higher than expected based on previous SBO efforts [5, 7, 12]. However, some of the plant response trends, such as the lack of natural circulation and timing of TDAFW pumps, were unclear. Future work should include a deeper look at the root cause of said trends, and the model will be updated accordingly based on additional insights.

4. Physics-Informed Physical Security Simulations

The end goal of this project is a proof-of-concept simulation involving a single site that demonstrates bidirectional communication between a force-on-force simulation and a physics simulation with exploration of key underlying uncertainties. The bidirectional communication is an additional level of realism that can reflect how adverse actions taken against a site can lead to necessary changes in tactics and strategy during a simulation by all teams involved (adversary, response force, and operators).

More concretely, several scenarios involving the Lone Pine Nuclear Power Plant will be modelled. The Scribe3D tabletop software will be used as a pilot software to model the force-on-force aspects, and the MELCOR severe accident software will simulate the physics of the nuclear reactor and transport of radionuclides. To aid in the exploration of the scenarios and their underlying uncertainties, the ADAPT dynamic event tree (DET) software will be used as the principle driver for the concurrent simulations of Scribe3D and MELCOR.

4.1 ADAPT

ADAPT is a Sandia-owned and developed software used for the generation and analysis of dynamic event trees (DETs) to evaluate the impact of uncertainties on the outcomes of simulations. The primary idea behind ADAPT, and DET analysis in general, is to assess the impact of uncertainties in simulations by generating all possible end states from all possible outcomes of key events determined to have occurred by the underlying simulation software. Key events can include anything: breaching of a fence-line, disabling a safety system, erroneous triggering of a safety system by wildlife, failure of a component due to wear, etc. Each of these events, if triggered, will lead to a different outcome as determined by parameters in the underlying simulation software (i.e., simulator), and those parameter values could influence the triggering of other key events in the future. From ADAPT's perspective, one event branches into several possible timelines each with their own values of the event parameters, and each of those timelines can lead to different sets of event occurrences that branch into even more possible timelines. This ripple effect of the interplay between these parameter values and the occurrence of key events determined by the simulator leads to the generation of a DET.

An example DET is shown in Figure 4.1. Each of the black boxes represents a timeline progressing forward in time (left-to-right) with a unique set of parameters and history. The arrows indicate when each of the timelines branch into multiple timelines, and no time changes during the creation of the branches regardless of the arrow's length (i.e., the start of the arrow is just prior to the event occurring, and the end of the arrow is just after the event has occurred). Several selected timelines have been given an identifier with all others are omitted for tidiness. The identifiers have the form of "TL" (timeline) appended with the event number and the branch number separated by a hyphen. All timelines after the first split have their parent's identifier appended with an interceding "|" (read as "given"). The times at the bottom correspond to the sequence of timelines ending with TL9-1 | TL7-1 | TL1-3 (highlighted in white).

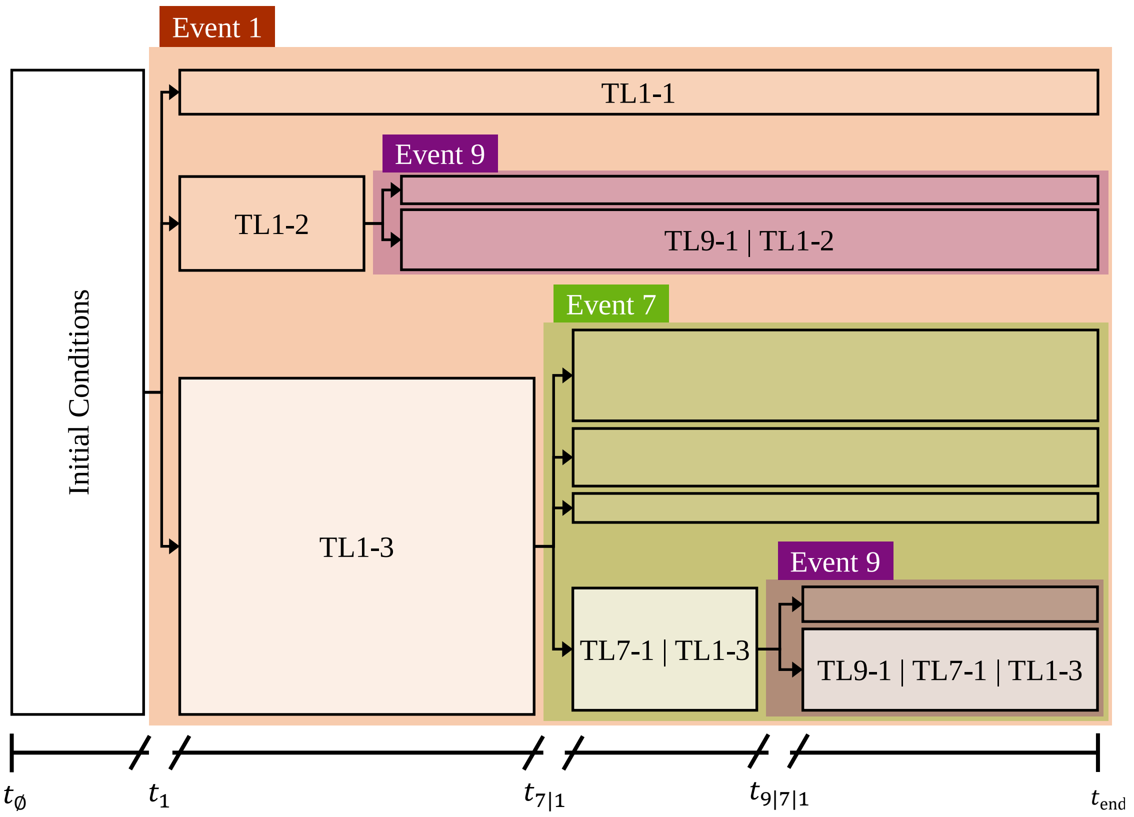


Figure 4.1 Example of a Dynamic Event Tree

The example dynamic event tree shows several key features. Every time an event occurs, the timeline branches into one or more timelines. Each of these timelines have their own set of parameter values that changes how the simulator behaves, which is represented by the branch number. Therefore, the only difference between TL1-1, TL1-2, and TL1-3 is the value of the parameters chosen by the analyst to distinguish the branches. Each of the events can spawn an arbitrary number of timelines, and the choice of three, four, and two splits for Events 1, 7, and 9, respectively, was for example purposes. As seen by Event 9 occurring twice in the example, events may repeat and may occur in different timelines and at different times as determined by the simulator and the timeline's history.

The subsections will discuss ADAPT at a high-level and the requirements for generating a DET like the one shown in Figure 3.1, the several improvements made since the most recent public release will be discussed, and lastly, the future work and needed improvements to ADAPT to effect the end goal of this project will be outlined.

4.1.1 Overview

ADAPT is a Python-based application designed for high-throughput computing, one-simulation per processor on thousands to millions of processors, for use with a simulator that satisfies a small number of requirements. A full description of ADAPT's usage and the source code itself is available at Reference [4], but a brief overview will be given here.

The general workflow for generating a DET with ADAPT is the following:

1. The simulator or simulators are added to ADAPT's list of registered simulators.

2. The user selects the simulators to use and provides ADAPT with several required files that define the DET
    a. a Python script called a wrapper that defines, among other things, how all simulators being used should be called, launched, and how to change their simulation parameters when branching; there is more discussion of the wrapper for this work in Section 5.
    b. a configuration script called the Branching Rules File that lists all the events that are part of the DET (as determined by the analyst), the number of branches that are generated when the event occurs, and a probability for each of those branches;
    c. and any other files that maybe required for the simulators to operate correctly.

3. With all the information from Step 2, ADAPT can begin the generation of the DET and does so by launching the first simulation with the initial conditions.[b]

4. When the simulator determines an event has or is about to occur:
    a. the simulator stops itself;
    b. ADAPT duplicates the inputs and rewrites their simulator parameters according to the Branching Rules File;
    c. ADAPT stores all the parent-child relationships and other pertinent data in its database;
    d. and ADAPT loads all child branches into its launch queue for the branches to continue their simulation with the modified parameters.

5. Step 4 continues until all child branches reach the final simulation time or the analyst requests earlier termination or pauses the process.

The number of potential timelines is limited only by the outcomes predicted by the simulator and the complexity of the Branching Rules File. The number of concurrent simulations that can occur is only limited by the amount of computational resources made available to the ADAPT scheduler; limitations and needed future work on this subject are discussed in Section 4.1.3.

There are additional details to this workflow for ADAPT, but those are left to the User Manual [4] and are not presented without loss of generality. However, given this workflow, there are three requirements that all ADAPT simulators must satisfy to be used with the system:

- Every simulator must have the ability to communicate to ADAPT that an event has occurred (e.g., breach of a fence-line or the disabling of a safety system); this is typically done by writing to an external data file that ADAPT can read.

[b] The formal term for DET generation in ADAPT jargon is *experiment*.

- Every simulator must also be able to stop its process the instant it determines an event occurs, permit some change in its simulation values by ADAPT, and be able to restart its process just after the event occurs using the adjusted simulation values.

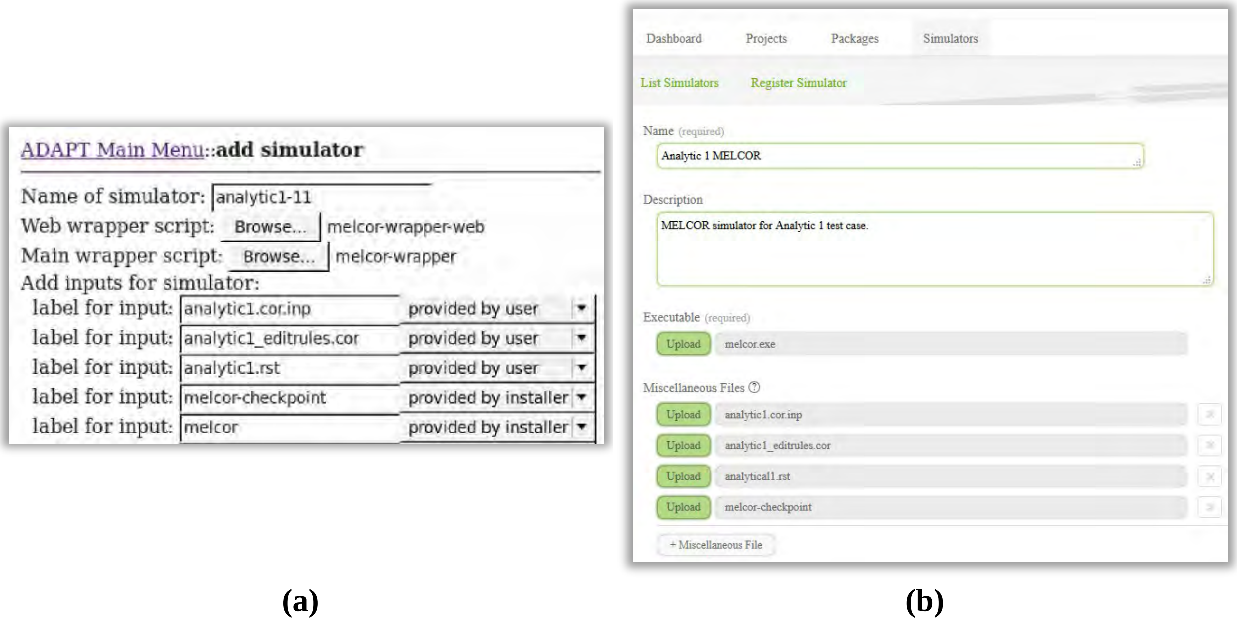
- Every simulator must be able to be stopped by ADAPT at any time during its process.

Both MELCOR and Scribe3D implement these features and are therefore compatible with the system. The primary effort needed for this project is to run the simulators in a concurrent or concurrent-like manner. The flexibility and simplicity of ADAPT does allow for this and will be further discussed in Section 5.

4.1.2 Improvements

The ADAPT database structure received an overhaul towards a more modular framework for declaring simulators and launching the generation of the DETs. The changes include the idea of Packages which are templates for DET generation with all simulators bound to the Package and an explicit list of required files the analyst must supply for DET generation to occur; older versions left these requirements ambiguous. Further, file handling was changed to allow for a deduplication of repeated simulators registered by analysts to the system for smaller and faster database calls. Also, a robust and expressive database manipulation system was developed to allow for more secure and less error-prone changes to the database. There were also several database entries added to facilitate extension to ADAPT for general use in high-performance computing scenarios that were not part of the original ADAPT structure.

In cadence with the database updates, the ADAPT user interface has old views upgraded and new features, like Packages, added. The changes in Simulator Registration are shown in Figure 4.2 for two equivalent simulator registrations. A setup like Figure 4.2 but using the new Package Interface is shown in Figure 4.3; the Branching Rules File (analytical1_editrules.cor) that was bound to the simulator in the old interface can now be declared as an analyst requirement when DET generation is started that is independent of the actual simulator (melcor.exe in the example).



Name of simulator: analytic1-11
Web wrapper script: melcor-wrapper-web
Main wrapper script: melcor-wrapper
Add inputs for simulator:
label for input: analytic1.cor.inp provided by user
label for input: analytic1_editrules.cor provided by user
label for input: analytic1.rst provided by user
label for input: melcor-checkpoint provided by installer
label for input: melcor provided by installer

(a)

Name (required)
Analytic 1 MELCOR

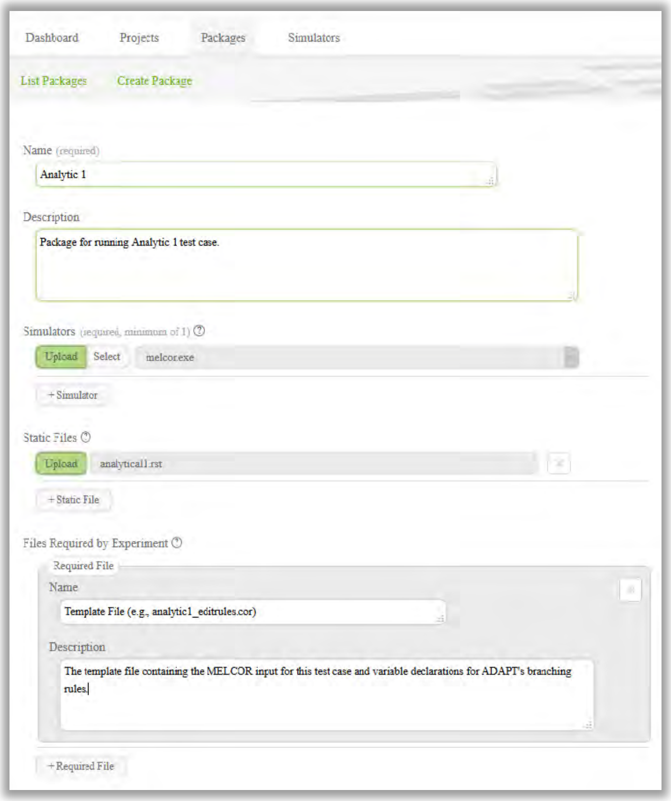
Description
MELCOR simulator for Analytic 1 test case.

Executable (required)
melcor.exe

Miscellaneous Files
analytic1.cor.inp
analytic1_editrules.cor
analytical1.rst
melcor-checkpoint

(b)

Figure 4.2 Comparison of ADAPT's Old Simulator Registration Interface (a) and the Updated Version (b)



Name (required)
Analytic 1

Description
Package for running Analytic 1 test case.

Simulators (required, minimum of 1)
melcor.exe

Static Files
analytical1.rst

Files Required by Experiment
Required File
Name
Template File (e.g., analytic1_editrules.cor)

Description
The template file containing the MELCOR input for this test case and variable declarations for ADAPT's branching rules.

Figure 4.3 Example of New Package Interface

4.1.3 Future Work

Originally, ADAPT was written to run on small-to-medium size computer clusters with no other job scheduling software other than ADAPT's internal queue system. Further, despite being primarily written in Python, ADAPT relied exclusively on Unix-like applications that implemented the Secure Shell and Secure Copy Protocols for launching queued simulations on remote machines. For this work, it will be necessary to extend these capabilities to use launching methods that are not bound to Unix-like applications such that ADAPT can run on Windows-based operating systems. Further, making ADAPT aware of other job scheduling software commonly found on modern clusters and being able to work in tandem with that software will be needed. These improvements will enable execution of large high-throughput calculations on a diverse array of computing clusters that will permit high resolution of all explored uncertainties across the scenarios.

5. Coupling ADAPT to SCRIBE-3D and MELCOR

ADAPT consists of several packages, including *ADAPT Server, Database, editrules* and *wrapper* files. *ADAPT Server* is the package that manages the backend of a DET. This includes the job-scheduling task for high performance computing (HPC) clusters, as well as transferring information from one branch to child branches. The *ADAPT Database* manages the data after it has been collected and can be interrogated to group data based on how the plant responded to specific points of uncertainty. In order link a new simulator (or combination of simulators) the user will need to create a new *wrapper* file. Additionally, to perform any ADAPT simulations, the user creates an *editrules* file which describes the uncertainties and how they are resolved.

At a minimum, in order to link a simulator to ADAPT the simulator must meet a set of basic requirements. These requirements are that any simulator must [4]:

- Stop on system values crossing a pre-defined threshold;
- Stop on command from ADAPT;
- Output the reason for any code stoppage, and;
- Restart using modified system parameters.

5.1 Linking

In order for ADAPT to use a simulator which meets these requirements, a *wrapper* file must be made. This *wrapper* contains simulator-specific instructions for ADAPT that are performed for each branch. At a minimum, this includes the instructions to execute the simulator or simulators for one branch of an experiment.[c] These simulators will have one of three possible outcomes:

- The simulator could stop at a branching condition;
- The simulator could reach the simulation end time for the experiment, or;
- The simulator could fail.

If the simulator stops due to a branching condition, it is required to report a code corresponding to the branching condition that was reached (see Section 4.1.1). This code is given to ADAPT and matched to an event in the *editrules* file to determine what variables need to change for child branches that are produced, and which simulator is to be the next executed. The necessary simulator files are handed off from the parent branch to its child branches and the wrapper ends. *ADAPT Server* then then performs the necessary job scheduling to execute the child branches; Figure 5.1 illustrates this process.

[c] “Experiment” is the term used by ADAPT to refer to a single DET analysis of one scenario.

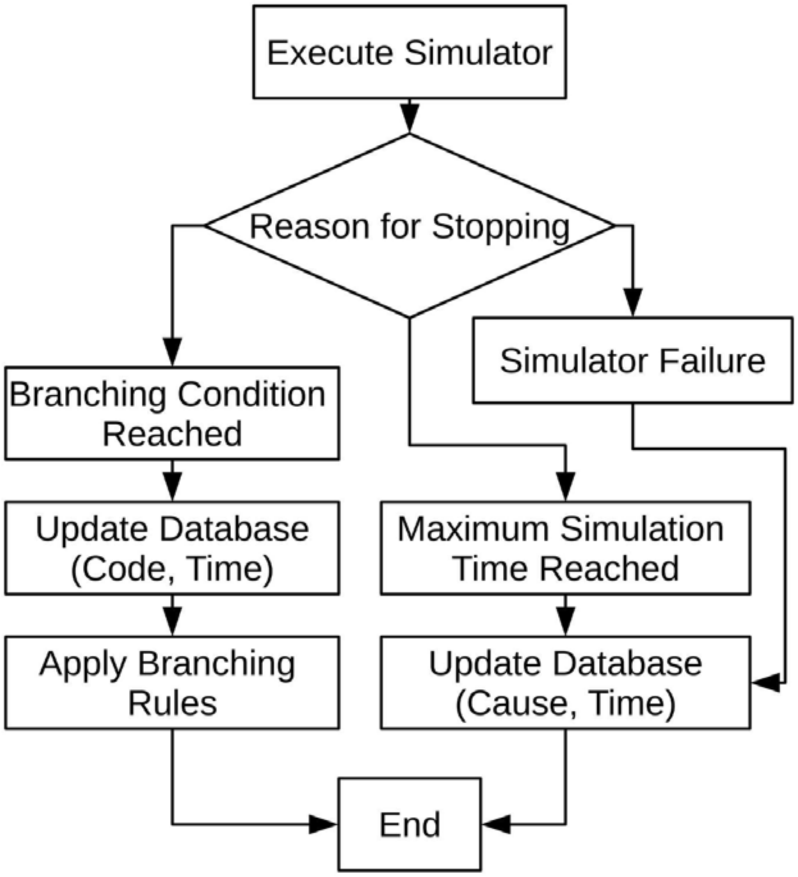

Execute Simulator
Reason for Stopping
Simulator Failure
Branching Condition Reached
Update Database (Code, Time)
Maximum Simulation Time Reached
Apply Branching Rules
Update Database (Cause, Time)
End

Figure 5.1 Minimum ADAPT Wrapper Actions [4]

5.2 Multiple Simulators

In 2016, the capabilities of ADAPT were expanded to allow multiple simulators to be linked to ADAPT in one experiment [9]. As a part of this change, the formatting of the *editrules* file was changed to require the analyst to specify which simulator to call initially, as well as after each instance of branching. For example, if ADAPT is being used to drive a force-on-force simulation, there may be a branching condition when adversaries detonate a breaching charge attached to a door. Upon reaching this condition, the *editrules* can specify that the next simulator to call is a shock physics model in order to determine the effects of the breaching charge. A flowchart illustrating the multiple simulator branching process is presented in Figure 5.2.

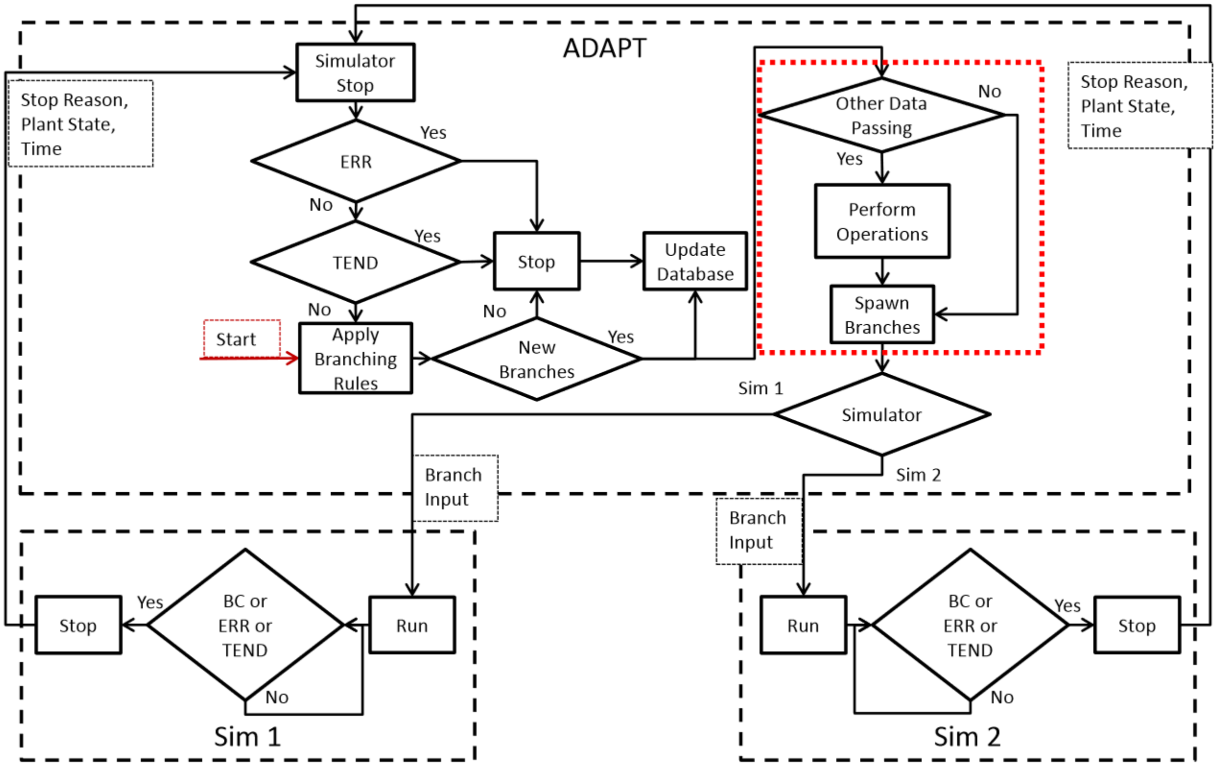


Figure 5.2 Branching Process for Two Generic Simulators [9]

5.3 Limitations with Multiple Simulators

One challenge with the way that ADAPT handles experiments with multiple simulators is that in the *editrules* for multiple simulators, one of the decisions made at a branching condition is choosing which simulator to call for the next branch. However, in a combined safety/security scenario it is not necessarily known which simulator will next need to be run. Therefore, applying this methodology to the safety-security (2S) case would lead to nonfunctional states. For example, if at a given time *t* a branching occurs in safety code, the next branching condition of the safety code might occur at $t + 10h$, but there might be a branching condition which would occur in the security code at time $t + 40m$. As branching conditions within either the safety or security models can have implications which affect both models, the branch that occurred in the security code at time $t + 40m$ may affect how the safety code runs past time $t + 40m$. If the safety code runs with the conditions at *t* until time $t + 10h$ without taking into account the branching of the security code at time $t + 40m$, many hours of computing time may be spent on the safety code analyzing an inaccurate condition (i.e., nonfunctional state) of the plant. Beyond the computational time that was spent analyzing a nonfunctional state, it may not be possible to recover the plant status; typically plant safety models are only saved when branching occurs to reduce unnecessary bloat in file sizes.

Therefore, to reduce the computational effort spent modeling inaccurate plant states and ensure availability of a restart files when needed, the 2S analysis will, instead of using ADAPT's current multiple simulator functionality, manually implement a method of branching adapted from the ADS-IDAC [10, 11] philosophy. In ADS-IDAC, the accident dynamics simulator (ADS) is directly linked with the Information, Decision, and Action in a Crew context cognitive model (IDAC) and are incorporated into the overall structure of the code. At every time step, the ADS model updates the physical status of the plant and passes the necessary information to the IDAC code, which models crew behavior. In this way, if either ADS or IDAC reach a branching condition, the other model receives that information immediately and can incorporate the necessary changes. The downside of the ADS-IDAC philosophy is that in order to transfer data between both models at each time step, it is a practical necessity for both models to be connected through memory, rather than exiting and transferring files. Because the models do not fully close when transferring between simulators or when undergoing branching, the simulation cannot be resumed if one branch were to fail, and the data generated up to that point would be lost. The necessity of transferring information through memory rather than through files also requires more effort developing the linkages between the simulators of choice, which increases the difficulty for users following this methodology creating links between their codes of choice.

Because the additional time needed to transfer files makes changing simulators at each time step impractical, it is planned within the proposed work to use of a hybrid system combining the ADS-IDAC approach and the ADAPT approach. This hybrid system will use short time blocks, of approximately 10 minutes in length. In this hybrid system, one model will be designated as the Leading Simulator (LS). The other model will be designated the Trailing Simulator (TS). Each ADAPT-generated branch will begin with execution of the LS, which operates until either it reaches a branching condition or the end of the current time block. During this simulation, the state of the LS will be saved at regular intervals. After the LS completes, the TS operate until either reaching a branching condition or the simulation time the LS ended.

If the TS completes its simulation time without reaching a branching condition, the LS will restart and all previous restart files will be deleted from the save memory. However, if the TS reached a branching condition during this time, the LS will be resumed from the nearest saved state to the simulation time the TS branching. The LS/TS methodology is illustrated in Figure 5.3. In this schematic, the scenario begins at time t=0 with the LS Branch Number 1 (BN_1). This simulation continues for a fixed time block before ending at $t=t_1$ without any branching conditions being met, which is represented by the diamond shape terminating the simulation. The TS, in BN_2, is then called at time t=0 to determine if this simulator reached any branching conditions before t_1. The diamond end cap shows that BN_2 TS did not and the LS is restarted at time t_1 with BN_3. Sometime before the end of this time block the LS reached a branching condition, marked by the star end cap. BN_4 was then called at time t_1 to determine if it would reach a branching condition before the time the LS did. As the BN_4 TS did not, the LS branching conditions are called which leads to BN_5, BN_6 and BN_7, beginning at the time the LS reached the branching condition. Following just BN_7, the LS did not reach a branching condition in the next time block, and BN_8 was again restarted from the beginning time of the most recent LS run. In this case, however, the TS has reached a branching condition during the time block, as marked by a star end cap. BN_8 has two child branches (BN_9 and BN_{10}) which were restarted from the LS at the time of the branching condition by the TS. Under the LS/TS framework, new branches will always be explored first by the LS and the TS will only play catchup, never extending its simulation time beyond that of the LS.

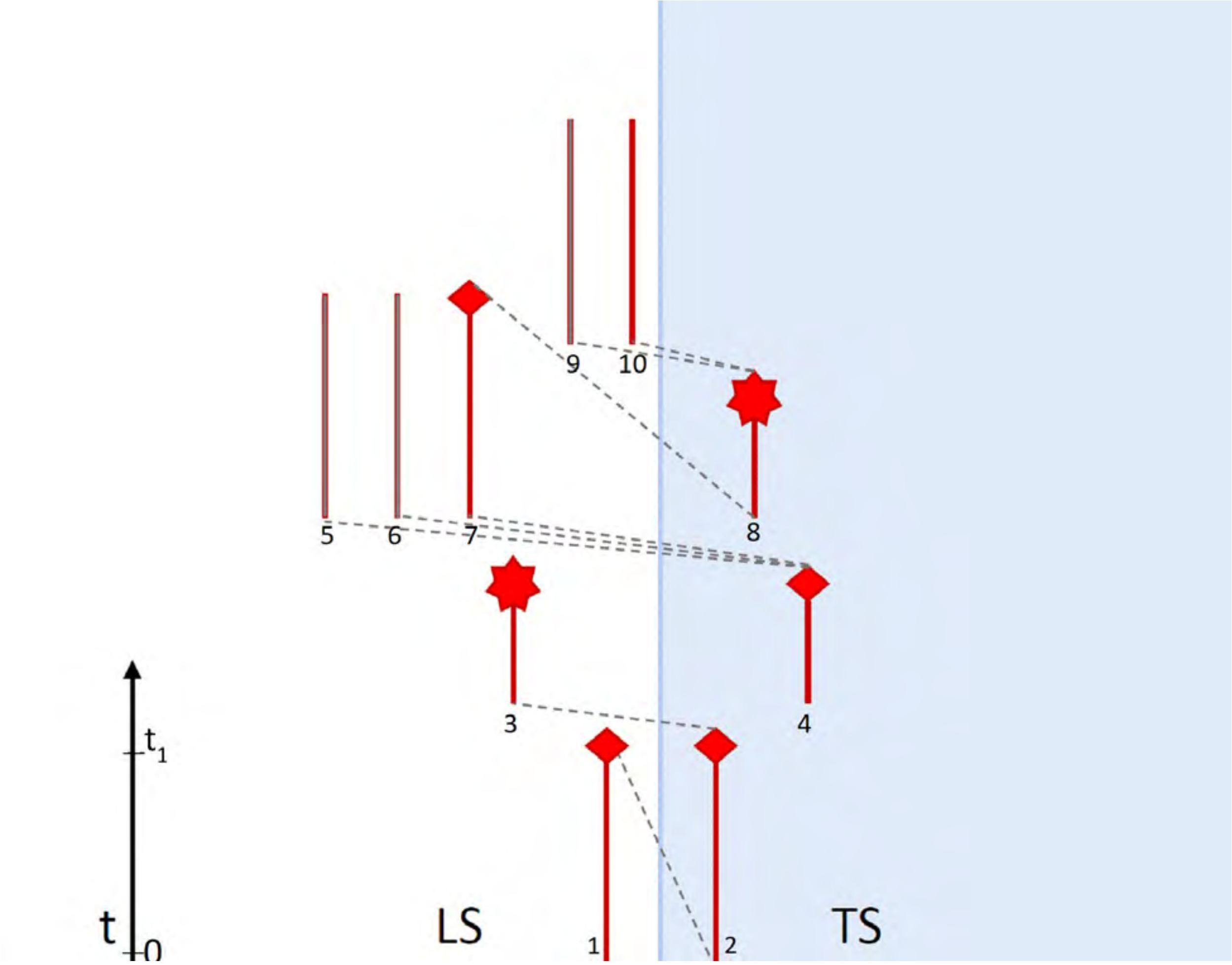


Figure 5.3 Schematic of the LS/TS Methodology

5.4 Linking with MELCOR

The 2S analysis is using two system models in conjunction with ADAPT. These are a safety system code, MELCOR (Section 3) and a security force-on-force code, Scribe3D (Section 2.7 and Section 5.5). Despite the severe accident capabilities in MELCOR, there exist some limitations. The initial code architecture predates the prevalence of multiple processors, and as such MELCOR uses only a single processor. Additionally, MELCOR simulations often run in real time, such that a 24-hour simulation will require approximately 24 hours of running time. Finally, during severe accidents, the physical phenomena are very complex and quickly changing.

As MELCOR has been used several times in ADAPT-based experiments, both as the sole simulator [12] and as one of multiple simulators [9], the code architecture is compatible with ADAPT. However, beyond this compatibility, individual MELCOR models intended for use with ADAPT need to be modified. For example, the behavior within a MELCOR model that stops execution upon reaching a branching condition is not inherent to the code. Instead, individual models need to have this behavior added, which can be a complicated process and has not been completed. In addition, variables within the MELCOR input file need to be altered to ADAPT-friendly versions.

To perform branching, ADAPT searches for specific placeholder variables in a template version of the *Save file*, which in MELCOR is its input file. Variables which are to be controlled by ADAPT are not given values within MELCOR. Instead, placeholder strings enclosed by variables are used. For example, if a given variable in MELCOR is to be handed over to ADAPT, it might be assigned the placeholder {V10035} rather than any meaningful value. In ADAPT, both the '{' and '}' characters are typically designated as the start and end characters of an ADAPT variable for MELCOR, which leads to this character string within MELCOR's input file being treated as a variable with name *V10035* that ADAPT is free to replace data as necessary. As the LS/TS framework is intended to replace ADAPT's default branching behavior for this analysis, the same system of modifying MELCOR input files is being used by the LS/TS framework. Instead of modifying MELCOR's *Save file*, ADAPT is directly linking to the LS/TS framework, which will contain a snapshot of the current values of all ADAPT-controlled variables. From here, it is the LS/TS framework that will input the correct variables (using the same methodology as ADAPT) into MELCOR's input file at the start of each branching event.

5.5 SCRIBE 3D

Scibe3D© is a Sandia Labs developed 3D scenario simulation tool used for scenario planning, execution, analysis, playback, and tabletop support; see Figure 5.4. It is a user-friendly tool that allows users to add personnel and vehicles to a fully realized 3D environment that then can build realistic, high-fidelity scenarios and play those scenarios back from multiple angles and speeds. Scribe3D is equipped with a combat calculator and detection system for realistic engagements, vehicle simulators (air and ground) for better movement/timelines, and modifiable effects such as fire, explosions, and plumes. It also has a simulation capability to run Monte-Carlo simulations on scenarios developed by a single-analyst or from a tabletop exercise. It has high-quality visualizations and flexibility allow it to support a multitude of different exercises and events.

The current LPNPP model is in Scribe3D. Section 2 provides an evolution of the LPNPP model and its final state of configuration within Scribe3D.

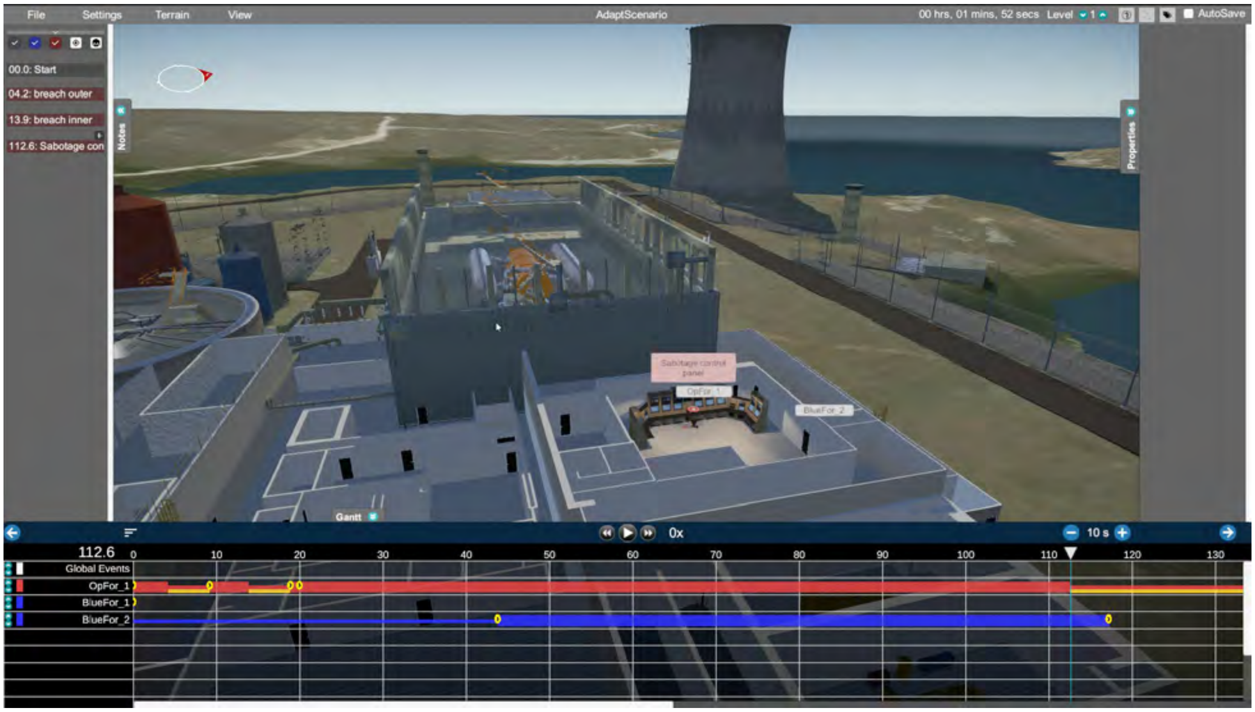
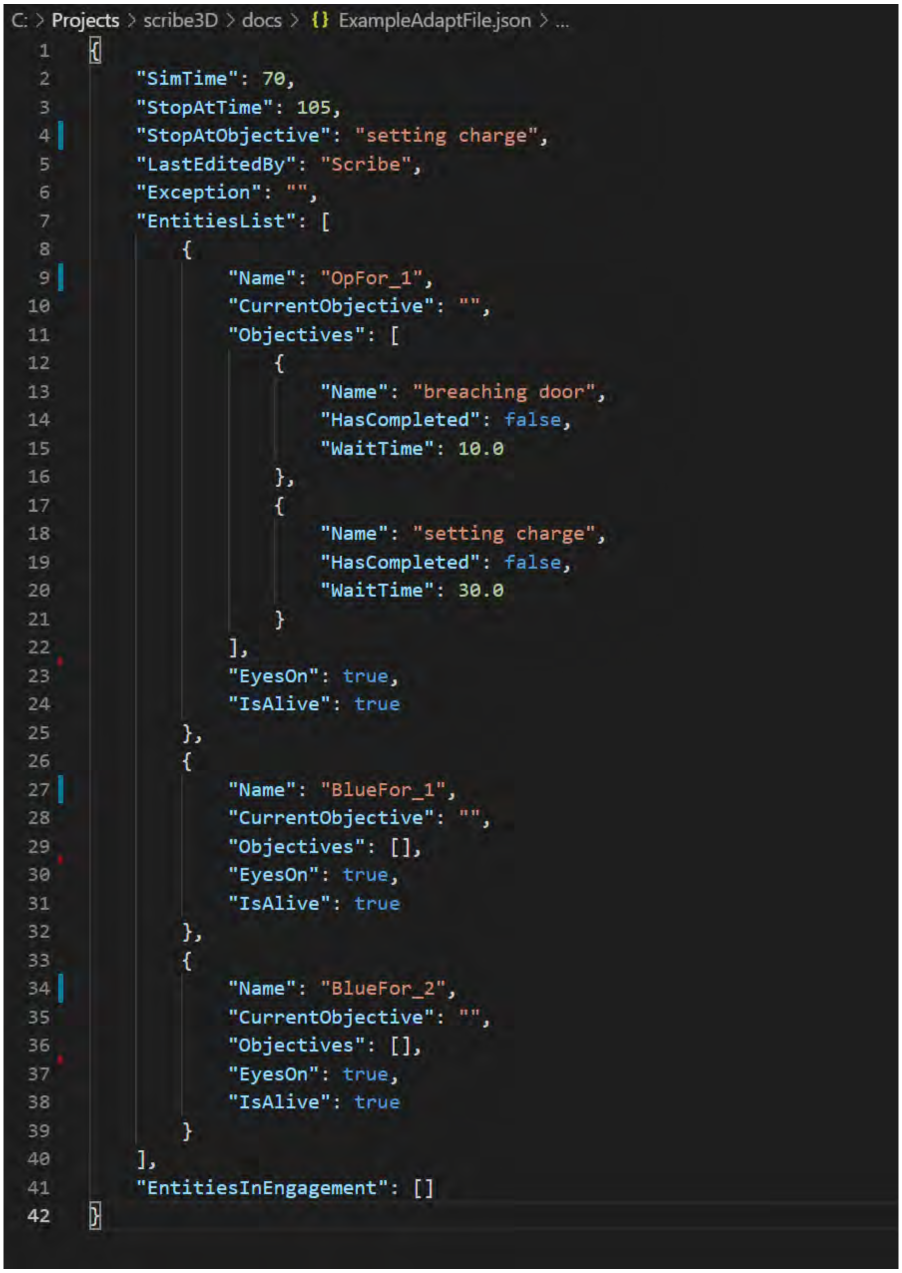


Figure 5.4 Scribe3D Main View

5.5.1 Scribe3D-ADAPT Integration

Scribe3D had to be updated and modified in a variety of ways to be able to link and run scenarios with ADAPT. The largest change was updating the Scribe3d simulation capability to work within the ADAPT paradigm. Scribe3D was initially developed with a capability to build a scenario and then run batches of simulations in a Monte-Carlo fashion. This was an effective tool for analysis and a good starting point but needed to be modified to work with ADAPT. Additionally, Scribe3D was developed to simulate and record the entire scenario and had to be entirely reworked to start at specific time within a scenario and then stop at a specific time (e.g., an adversary or response force individual achieved a specific objective or after an engagement). The recorded simulation then had to be stitched into the scenario and saved into a different file.

Scribe3D also needed to be able to notify ADAPT on the scenario changes and to make changes in the scenario from ADAPT input. This is done by writing to a file in *JSON* format. This file is used to set the time, stop parameters, results of engagements, and more. An example is provided below in Figure 5.5 with a description of each field.



C: > Projects > scribe3D > docs > {} ExampleAdaptFile.json > ...

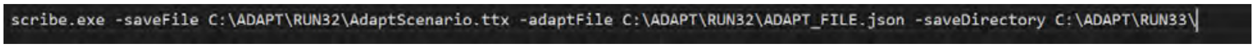
```
{
    "SimTime": 70,
    "StopAtTime": 105,
    "StopAtObjective": "setting charge",
    "LastEditedBy": "Scribe",
    "Exception": "",
    "EntitiesList": [
        {
            "Name": "OpFor_1",
            "CurrentObjective": "",
            "Objectives": [
                {
                    "Name": "breaching door",
                    "HasCompleted": false,
                    "WaitTime": 10.0
                },
                {
                    "Name": "setting charge",
                    "HasCompleted": false,
                    "WaitTime": 30.0
                }
            ],
            "EyesOn": true,
            "IsAlive": true
        },
        {
            "Name": "BlueFor_1",
            "CurrentObjective": "",
            "Objectives": [],
            "EyesOn": true,
            "IsAlive": true
        },
        {
            "Name": "BlueFor_2",
            "CurrentObjective": "",
            "Objectives": [],
            "EyesOn": true,
            "IsAlive": true
        }
    ],
    "EntitiesInEngagement": []
}
```

Figure 5.5 Example of *Save File* for ADAPT

- SimTime: Time in seconds to start the simulation at
- StopAtTime: Time in seconds to stop the simulation at
- StopAtObjective: Name of objective to stop simulation at (stops at beginning)
- LastEditedBy: Application that most recently edited this file (either Scribe3D or ADAPT)
- Exception: Empty unless error occurred
- EntitiesList: List of all entities and their objectives
    - Name: Entity Name
    - Current Objective: Entities current objective
    - Objectives: Full list of entities objectives
        - Name: Objective name. Used in StopAtObjective field

        - HasCompleted: If the objective is complete. Can be set by ADAPT to skip objectives
        - WaitTime: Time to complete objective
    - EyesOn: If false, detection is not run for entity, preventing engagements
    - IsAlive: Sets whether entity is KIA (killed in action)
- EntitiesInEngagement: List of entities who were in an engagement during the last simulation Used to notify ADAPT of engagements.

Another modification is that Scribe3D needed to be started from the command line from ADAPT, load a specific scenario, modify it based on the ADAPT *JSON* file, and then run the simulation. This is all done through command line arguments; an example is shown below.



```
scribe.exe -saveFile C:\ADAPT\RUN32\AdaptScenario.ttx -adaptFile C:\ADAPT\RUN32\ADAPT_FILE.json -saveDirectory C:\ADAPT\RUN33\
```

- -saveFile: path of Scribe scenario to load
- -adaptFile: path of JSON parameters file
- -saveDirectory: path of directory to save results in

Scribe3D also needed a rework of some of its underlying structure to work with ADAPT. Previously, the event system within Scribe3D was tightly coupled with its waypoint system. This coupling can be advantageous while building scenarios but made things very difficult when recording simulations, especially when stitching simulated sections into a scenario. Decoupling these systems became necessary to integrate ADAPT with Scribe3D.

6. Joint Safety-Security Scenarios

The purpose of physical security is to ensure that adversaries are unable to effect sabotage or theft against the Lone Pine NPP, either by stealing nuclear materials which may be weaponized as either a radiological dispersion device or an improvised nuclear device or by damaging systems within the nuclear plant that would cause a release of radionuclides. Theft can only occur at locations where nuclear materials are present, but sabotage can take two forms: direct and indirect.

Direct sabotage is targeted specifically at a location with radionuclides with the intent of releasing those materials. Indirect sabotage includes sabotage of systems that are necessary for providing necessary cooling to reactor fuel in the core and the spent fuel pool in order to prevent a release. A subset of systems adequate to provide this cooling are protected as vital areas.

This section provides a reference on the assumed capabilities of adversaries that physical security forces are responsible for defeating and two sabotage scenarios.

6.1 Design Basis Threat

A hypothetical design basis threat (DBT) was created as part of the Lone Pine model update and is described Reference [3]. The DBT provides a broad level of adversary capabilities and, for specific scenarios, can be altered appropriately. The DBT considered for this work's scenario development is shown in Table 17-2 of Reference [3] for the Lone Pine NPP. Table 17-2 describes the intent, capabilities, equipment, and motivation of adversaries.

6.2 Vital Areas and Sabotage Scenario Generation

All sabotage scenarios necessarily require adversaries to either directly damage nuclear materials or damage safety systems that are required to remove decay heat from the reactor fuel and the spent fuel pool. To that end, it is necessary for all credible scenarios to, at minimum, sabotage systems located in specific rooms of the LPNPP. One combination of these areas that, if protected, are sufficient to prevent the release of radionuclides are designated the Vital Areas (i.e., target sets). While it is generally not known how adversaries are attempting to sabotage the LPNPP, protecting the Vital Areas ensures that the plant, as a whole, can achieve safe shutdown and provide sufficient decay heat removal.

Vital Areas are often determined from the Level 1 probabilistic risk assessment (PRA) performed for safety analysis. The Vital Area Identification (VAI) process takes the base components that are included in Level 1 PRA and combines them based on location (often by room); this is often called target set analysis when applied at domestic NPP facilities. VAI then uses Boolean algebra to construct a sabotage area logic model which determines the combinations of locations that, when sabotaged, lead to core damage. An example model to disable one auxiliary feedwater train is given in Figure 6.1. All of the minimal cut sets (i.e., unique combinations of locations with no combination containing all locations) that lead to core damage are collected for a VAI. The Boolean complement of these minimal cut sets are protection sets, or candidate vital area sets; NRC and utilities call these target sets. These contain at least one location from every sabotage cut set such that if all of these locations are protected, no cut set in the sabotage area logic model can be achieved by adversaries. One of these candidate vital area sets is selected by the facility to be the Vital Area set that is protected using a NPP's security posture.

To generate security scenarios, the sabotage area logic model was used to identify areas that adversaries could target. After finding a set of targets for the adversary, brainstorming determined possible strategies for adversaries to reach these targets.

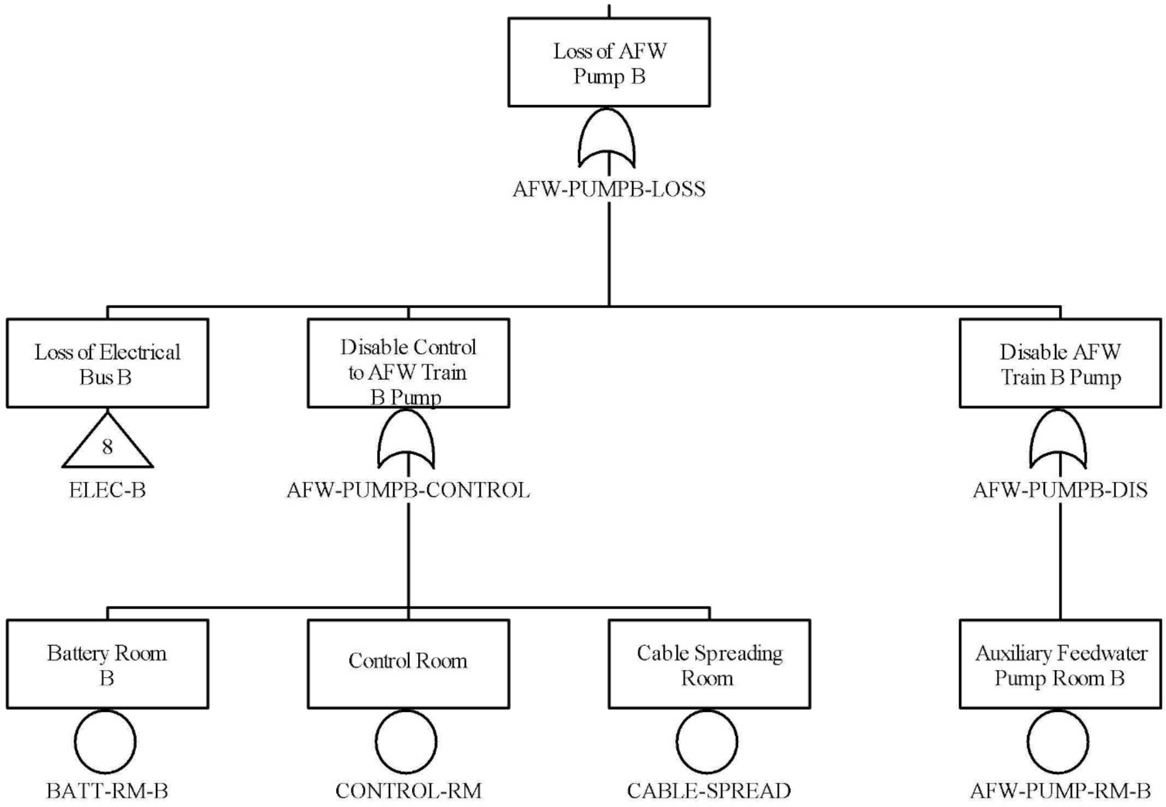


Figure 6.1 Extract from Sabotage Area Logic Model

6.2.1 Loss of Ultimate Heat Sink

This scenario does not use an insider. Instead, marksmen are added to the DBT with long-range rifles. Additionally, a boat is used instead of land vehicles.

The marksmen (Team RED 1), find a location with line of sight to the Lone Pine protected area, and particularly the guard towers and set up before the scenario begins. The remaining adversaries (Team RED 2) begin the scenario piloting a boat in Lake Winowich. Additionally, remote charges are pre-placed on the power lines leading out of the Lone Pine NPP facility.

At the beginning of the scenario, RED 1 neutralizes the guard in tower T4. Simultaneously, RED 2 approaches the intake structure and enters it, crossing the PIDAS, if necessary; see Figure 6.2.

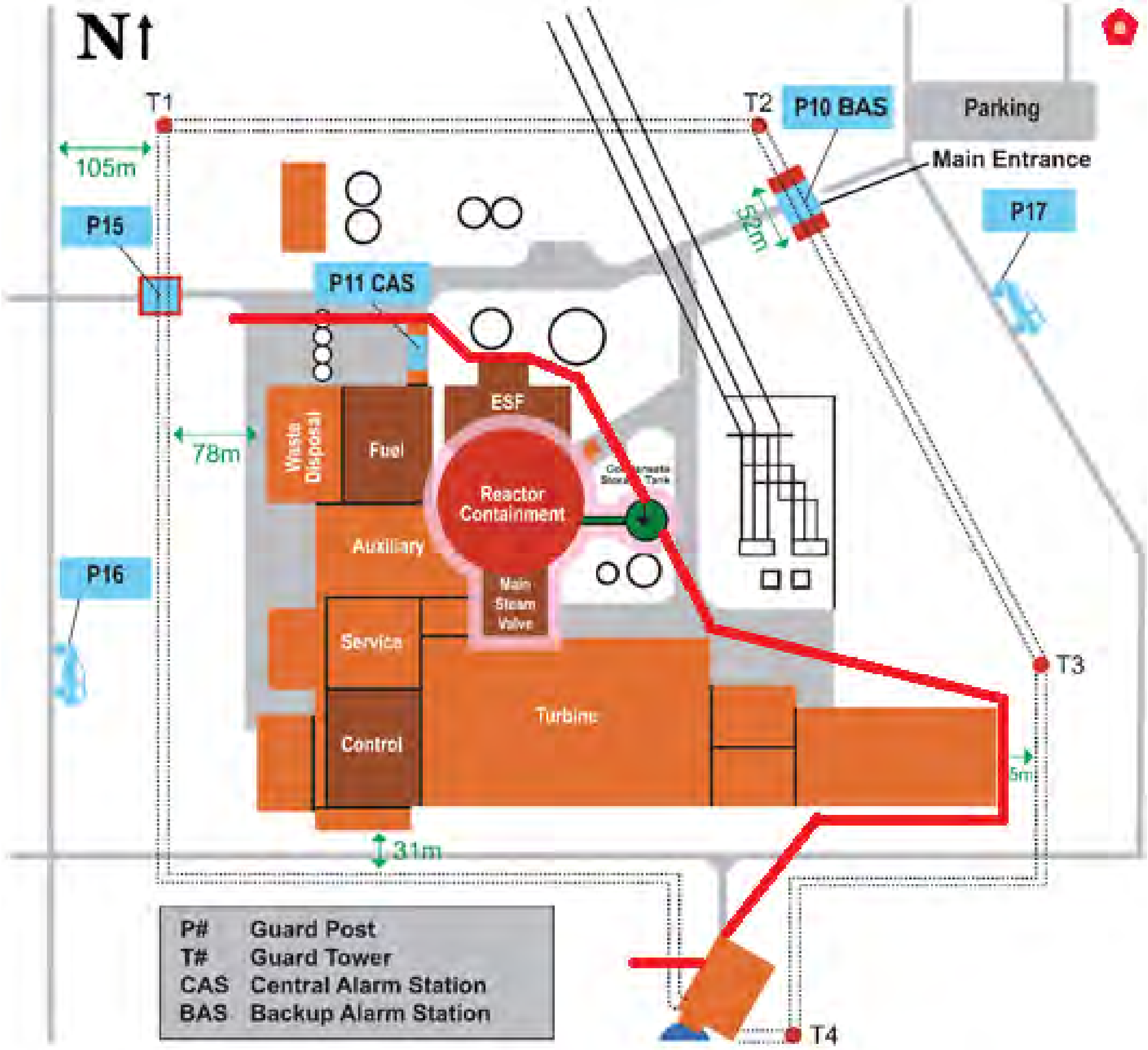


Figure 6.2 Illustration of the Loss of Ultimate Heat Sink Scenario

RED 2 enters the intake structure and destroys the pumps supplying water to the plant. RED 1 detonates the remote charges on outside power lines and then begins neutralizing guard towers, response force patrols, and other targets of opportunity.

RED 2 leaves the intake structure and crosses the protected area to the condensate storage tank (CST) under the protective cover of RED 1. RED 2 engages and defeats the response force with the assistance of long-range fire from RED 1, if necessary.

Upon arriving at the CST, RED 2 plants explosives on the wall of the tank, retreats to a safe distance and detonates; this explosion releases the water inside. If at least 3 members of RED 2 are still active, RED 2 then proceeds to the FLEX equipment building and destroys the stored equipment.

In either case, RED 2 enters the auxiliary building and interdicts travel by operators through the facility while RED 1 conducts overwatch of the protected area and interdicts activities outside. When offsite responders arrive, RED 1 notifies all members of RED 2 and they exfiltrate the facility on foot.

Figure 6.2 shows the initial adversary scenario, with RED 1 represented by a red pentagon in the upper-right corner and RED 2's pathway represented by the red line.

The response force initially attempts to neutralize the attacking adversaries. If unsuccessful, and the CST is damaged, operators attempt to inject as much water from the CST into the core as possible, releasing coolant into the containment sump if necessary, and uses feed and bleed cooling from the refueling water storage tank (RWST) after the CST empties. However, a maintenance error may have left the RWST unable to properly align to the high-pressure injection pumps. In this event, operators will either depressurize the reactor to use low-pressure injection or manually realign the RWST tank.

If operators find it necessary to manually realign the RWST tank, they will need to perform a field action in the auxiliary building before initiating feed and bleed cooling. In this case, operators will call for a member of the response force (Team BLUE 1) to provide an escort. BLUE 1 and an operator will travel together from the control room to the auxiliary building to perform the necessary actions. If the operator is killed by either RED team, no operator will leave the control room until the entire facility has been cleared of adversaries.

Additionally, offsite forces will arrive at Lone Pine to provide reinforcements and neutralize the adversaries. After these offsite forces arrive, they will spread out throughout the LPNPP structure and clear the building of adversaries. Once the offsite forces have made the all-clear determination, operators will begin setting up FLEX equipment; if it has not been sabotaged. Additionally, after 24 hours it is assumed that SAFER FLEX equipment has arrived from offsite and will provide cooling to the plant as necessary.

6.2.2 Vehicle Bomb

In this scenario, adversaries are divided into two teams. Team RED 1 consists of members driving a vehicle with a specially-built ramp. Team RED 2 consists of a 4-wheel drive truck that contains the equivalent of 200 kg of TNT in explosives.

At the beginning of the scenario, RED 1 drives to the PIDAS boundary and cuts through the chain link fence fabric. RED 1 then drives into the PIDAS and installs the ramp on their vehicle to the vehicle barrier along the inner fence of the PIDAS.

After the ramp has been installed, RED 2 drives their vehicle up the ramp and launches over the vehicle barrier, landing inside the protected area.

RED 1 travels on foot to the containment building while RED 2 drives to the nearest wall of the containment structure. Upon arriving at the containment building, RED 2 exits their vehicle and retreats to a safe distance before detonating the explosives onboard their vehicle. This detonation punctures a hole through the containment wall sizeable enough for humans to pass through.

RED 1 rendezvous with RED 2, and the combined force enters the containment structure through the newly-created hole. Adversaries then plant explosive charges on the lower head of the reactor pressure vessel, leave the containment structure, and detonate the explosives, puncturing the reactor vessel. This puncture yields an equivalent large break loss of coolant accident.

7. Summary

The Lone Pine Nuclear Power Plant (LPNPP) facility was updated in Scribe3D with full details of the LPNPP systems found in Reference [3]. Within in Scribe3D, the LPNPP facility represents the two-loop pressurized water reactor (PWR), support systems, the secondary systems to include the steam turbine, the cooling water systems, and overall security layout of the facility; see Figure 2.11 through Figure 2.17.

The Scribe3D software is the security force-on-force scenario development simulation tool used for this effort. Scribe3D was used for scenario planning, execution, analysis, playback, and tabletop support of the LPNPP attack scenarios. Scribe3D allows users to add personnel and vehicles to a fully realized 3D environment for realistic, high-fidelity scenarios and allows those security scenarios to be played back from multiple angles and speeds; Scribe3D is equipped with a combat calculator and detection system for realistic engagements, vehicle simulators (air and ground) for better movement/timelines, and modifiable effects such as fire, explosions, and plumes. Using Scribe3D's Monte-Carlo capability, security scenarios were developed for analysis with a linked reactor system response model in MELCOR.

For an evaluation of the reactor system response modeling using MELCOR, a short-term station blackout (STSBO) was considered. The STSBO MELCOR simulation predicted a quick core degradation transient with core damage occurring within an hour of the initiating event. The environmental release fractions for the scenario were higher than expected based on previous SBO efforts [5, 7, 12]. However, some of the plant response trends, such as the lack of natural circulation and timing of TDAFW pumps, were unclear. Future work should include a deeper look at the root cause of said trends, and the model will be updated accordingly based on additional insights.

The integration of safety (MELCOR) and security (Scribe3D) software through a dynamic event tree framework (ADAPT) has yielded some unique research challenges. Namely, the ability to link software within a Unix environment. This effort was not successful in linking within a Unix environment and has resulted in the recoding of the dynamic event tree framework (ADAPT) within a Windows environment. Updating ADAPT for Windows cluster job scheduling, will enable execution of large high-throughput calculations on a diverse array of computing clusters that will permit high resolution of all explored uncertainties across the safety-security scenarios. This coding effort and subsequent proof-of-concept of real-time linking of uncertain safety and security scenario data resulting in a dynamic event tree analysis will be completed by December 2019.

8. List of References

1. Pacific Gas & Electric Company, "PG&E Company 2018 Nuclear Decommissioning Costs Triennial Proceeding Prepared Testimony – Volume 1," December 13, 2018. https://analysis.nuclearenergyinsider.com/pge-seeks-decommissioning-head-start-cost-estimates-rise

2. Nuclear Energy Institute, "Delivering the Nuclear Promise," 2016-2019. https://www.nei.org/resources/delivering-the-nuclear-promise

3. Osborn, D., et al., "Lone Pine Nuclear Power Plant Description," SAND2019-12227, Sandia National Laboratories, Albuquerque, NM, 2019.

4. Sandia National Laboratories, "ADAPT - Dynamic Event Tree Generation and Analysis," National Technology and Engineering Solutions of Sandia, LLC., 2019. [Online]. Available: http://www.sandia.gov/adapt/

5. Sandia National Laboratories, "State-of-the-Art Reactor Consequence Analyses Project Volume 1: Peach Bottom Integrated Analysis," NUREG/CR-7110 Vol. 1, Nuclear Regulatory Commission, Washington, DC, 2012.

6. R.O. Gauntt, et al., SAND2012-6173, "Fukushima Daiichi Accident Study (Status as of April 2012)," Sandia National Laboratories, Albuquerque, NM, 2012.

7. Sandia National Laboratories, NUREG/CR-7110 Volume 2, "State-of-the-Art Reactor Consequence Analyses Project Volume 2: Surry Integrated Analysis," Nuclear Regulatory Commission, Washington DC, 2012.

8. R.O. Gauntt, et al., SAND2015-6692R, "MELCOR Computer Code Manuals, Volume 2: Reference Manuals, Version 2.1," Sandia National Laboratories, Albuquerque, NM, 2015.

9. Jankovsky, Z., M.R. Denman, and T. Aldemir, "Extension of the ADAPT Framework for Multiple Simulators," Transactions of the American Nuclear Society, Las Vegas, NV, 2016.

10. Chang, Y.H.J., and A. Mosleh, "Cognitive Modeling and Dynamic Probabilistic Simulation of Operating Crew Response to Complex System Accidents Part 5: Dynamic Probabilistic Simulation of the IDAC Model," Journal of Reliability Engineering and System Safety, pages 1076-1101, 2007.

11. LaChance, J, et al., "Discrete Dynamic Probabilistic Risk Assessment Model Development and Application," SAND2012-9346, Sandia National Laboratories, Albuquerque, NM, 2012.

12. Osborn, D., "Seamless Level 2 / Level 3 Probabilistic Risk Assessment Using Dynamic Event Tree Analysis, Dissertation, the Ohio State University, Columbus, Ohio, 2013. https://etd.ohiolink.edu/pg_10?::NO:10:P10_ETD_SUBID:5611

(This Page Intentionally Left Blank)