Automated Fire PRA Scenario Modeling in SAPHIRE Using FRI3D

Steven Prescott[a], James Knudsen[b], and Stephen Ted Wood[c]
[a] Idaho National Laboratory, Idaho Falls, ID, USA, steven.prescott@Inl.gov
[b] Idaho National Laboratory, Idaho Falls, ID, USA, james.knudsen@Inl.gov
[c] Idaho National Laboratory, Idaho Falls, ID, USA, stephen.wood@Inl.gov

Abstract: Current fire modeling practices require many manual steps and processes to transfer fire scenario information and data between software tools such as the Computer Aided Fault Tree Analysis System (CAFTA), the Consolidated Model of Fire and Smoke Transport (CFAST), and other databases. The Fire Risk Investigation in 3D (FRI3D) software was developed to help minimize these manual steps by automating as many steps as possible, and linking the 3D spatial information with the probabilistic risk assessment (PRA) information. Initially, FRI3D was coupled directly with the existing FRANX fire input data and CAFTA to perform fire analysis. The modular design allows for coupling with other PRA tools, and the PRA software Systems Analysis Programs for Hands-on Reliability Evaluations (SAPHIRE) was incorporated for a facility pilot project. SAPHIRE does not designate tools that can be assigned to specific fire scenarios. Fire scenarios must be added manually, which is time-consuming and potentially error prone. To make FRI3D compatible with SAPHIRE and simplify scenario modeling, a new module was created which automatically generates the fire scenario and inputs it into SAPHIRE for evaluation using SAPHIRE macros. This module constructs the PRA components, following a standard currently used to evaluate fire scenarios within SAPHIRE. This paper outlines the SAPHIRE fire modeling standard and the methods used by FRI3D, and provides an example of a fire scenario being added into SAPHIRE for evaluation based on a generic compartment from a generic facility.

1. INTRODUCTION

The Fire Risk Investigation in 3D (FRI3D) software implements and combines approved methods to simplify and automate many tasks required for nuclear power plant (NPP) fire modeling [13] [14]. The first phase of the development focused on coupling the most commonly used tools and methods used by the NPP industry, such as Electric Power Research Institute's (EPRI) Computer Aided Fault Tree Analysis System (CAFTA) software. However, not all nuclear facilities use CAFTA for PRA modeling, making it impossible to take advantage of the features provided by FRI3D.

As part of a future vision, FRI3D was designed in a modular fashion, making it easy to add other tools and methods. The PRA tool Systems Analysis Programs for Hands-on Reliability Evaluations (SAPHIRE) was included in the pilot study as an option for solving FRI3D-generated fire scenarios.

SAPHIRE is similar to CAFTA, as it is used to construct and calculate NPP Standardized Plant Analysis Risk (SPAR) models, typically for regulatory use. More recent updates to SPAR models have included a standard process for fire modeling aspects. FRI3D duplicated this process to automatically create and calculate its scenarios in SAPHIRE. This research outlines the SAPHIRE fire modeling process and how it is automated using FRI3D, which drastically reduces costs for development and maintenance at applicable facilities using fire PRA models.

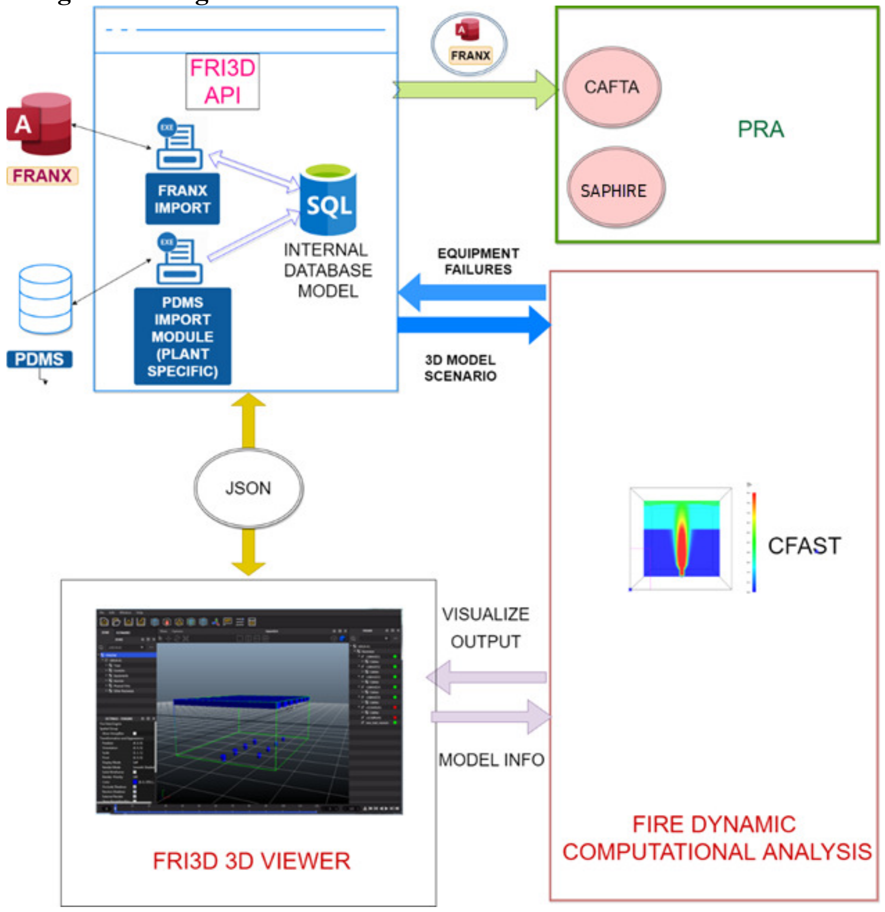
2. FRI3D CAPABILITES

The design goal of FRI3D is to integrate the key aspects of fire-PRA modeling, namely the PRA-logic model, spatial model, and fire simulation, into a single easy-to-use platform. This allows users to perform fire modeling, as outlined in NUREG/CR-6850 [1].

2.1 Fire Model Pieces

There are several aspects of a fire model that FRI3D maintains and links together. The software can import a plant's existing FRANX fire-PRA model and plant equipment databases such as Plant Data Model System (PDMS). FRI3D currently couples with Consolidated Model of Fire and Smoke Transport (CFAST) [2]for fire simulations with future plans to add Fire Dynamic Simulator (FDS) [3]. The PRA tools currently coupled are CAFTA [4] and as part of this research, SAPHIRE [5]. The interactions between tools are shown in Figure 1. For more information on this process, see [6].

Figure 1: Integration and Data Flow for the Different Areas of FRI3D

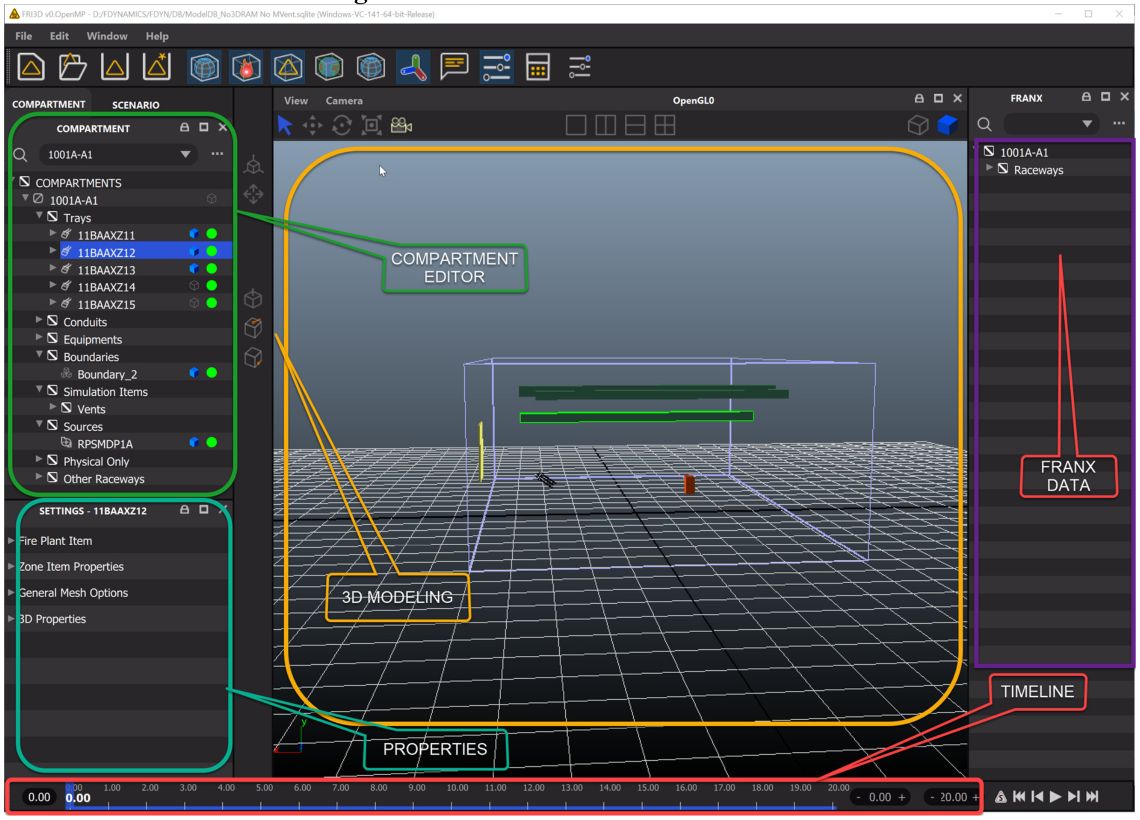


After initial setup, importing, generating, and running the different models or data is done automatically or through simple user menus. Fire logic mapping, simulation results, scenarios, 3D data, and PRA links are all stored in an internal database.

2.2 User Interface

FRI3D provides a graphical user interface that includes a simple CAD modeling tool. Here, the user can view different compartments and associated scenarios, then assign or modify the spatial information in the center 3D area. A properties window shows and allows the user to easily view and assign data (Figure 2).

Figure 2: FRI3D User Interface Features



2.3 Fire Scenarios

Fire scenarios can be manually developed, or automatically, by running a fire simulation. This eliminates the need for the various stages of hand calculations and manually assigned failed components. The following steps are used to auto-generate a scenario.

1. Construct CFAST model from FRI3D's model, which includes 3D data and properties.
2. Simulate CFAST using heat release rate (HRR).
3. Determine if there are any secondary combustibles using CFAST results and the FLASH-CAT [7] Method. If so, calculate a new HRR and go back to step 2.
4. Use simulation results to determine additional cable failures using the Thermally-Induced Electrical Failure THIEF [8] Method if cable data exists. If not, use heat soak method [9]. Also, use simulation data to determine direct component failures for the compartment.
5. Use the fire logic mapping to determine subsequent components that failed due to cable failures.
6. Save failures, timing, and other data as a scenario for the compartment.

Scenarios generated can then be evaluated or modified by the user and sent to the PRA model for a conditional core damage frequency calculation.

3. SAPHIRE FIRE MODELING STANDARD

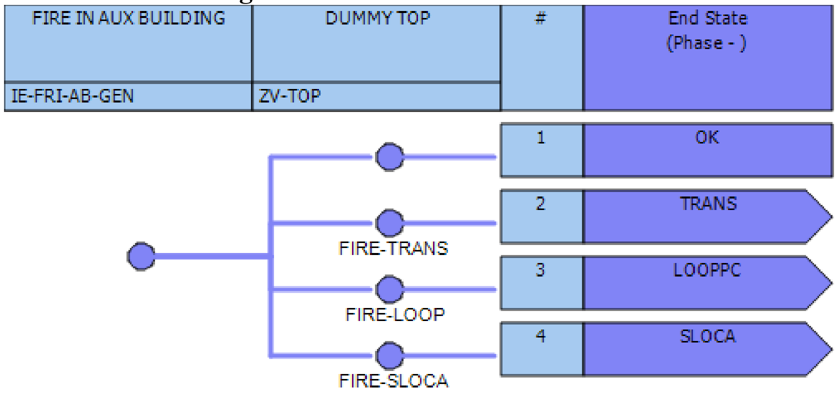
There are multiple ways to incorporate fire scenarios into a PRA model, such as adding them directly into fault trees or into event trees. Scenarios can be incorporated directly into existing internal events PRAs. There are different types of PRA modeling software such as RISKMAN, CAFTA, and SAPHIRE. Each has different features that allow for the creation of fire scenarios. This section goes over a standard used at Idaho National Laboratory (INL) for the SPAR modeling. The fire modeling descriptions assumes the reader has a good understanding of the SAPHIRE modeling process, including event trees, fault trees, rules, and change sets. For more information on SAPHIRE modeling, refer to the SAPHIRE User's Guide [10] or training information [11].

3.1 Fire Scenario Event Tree

The fire scenario event tree is a generic starting point for fire modeling. The event tree can be complex with multiple branches for all possible conditions, as shown in Figure 3, or simplistic with a single branch event tree, based on specific conditions of the fire. The primary function of the fire scenario event tree is to provide a place for the initiating event frequency and establish the conditions created by the fire. Therefore, the event tree must be capable of handling fires that may induce a plant loss of offsite power (LOOP) event, which may induce a loss of coolant accident (LOCA) due to spurious operation of valves, or a standard plant trip.

Figure 3 provides an illustration of a complex fire scenario event tree with a branch and a transfer point to three primary conditions that occur given a fire. The top branch transfers to the generic transient event tree conditioned on the fire, only causing a reactor scram. The second branch transfers to the LOOP event tree, which is conditional on the fire causing a LOOP. Lastly, the third branch transfers to a LOCA event tree conditioned on a spurious operation of a valve, which leads to a LOCA.

Figure 3. Fire Scenario Event Tree



The fire scenario event tree can be pruned down to the specific condition of the fire if necessary; however, using a slightly more complex event tree allows for all conditions to be evaluated. A majority allow for the development of explicit conditions (e.g., LOOP).

The top event of the event tree represents the fire scenario frequency. The next top event contains the specific logic that is used to establish the conditions of the fire scenario, FIRE-TRANS, FIRE-LOOP, and FIRE-SLOCA. For example, the fault tree FIRE-LOOP contains the offsite power sources that a fire scenario could render failed. Therefore, this fault tree would be conditioned as occurring (i.e., TRUE), and then the fire scenario would transfer to the existing LOOP event tree to evaluate the fire scenario.

Once the generic event tree has been developed, the next part of fire modeling within SAPHIRE is to identify the target sets.

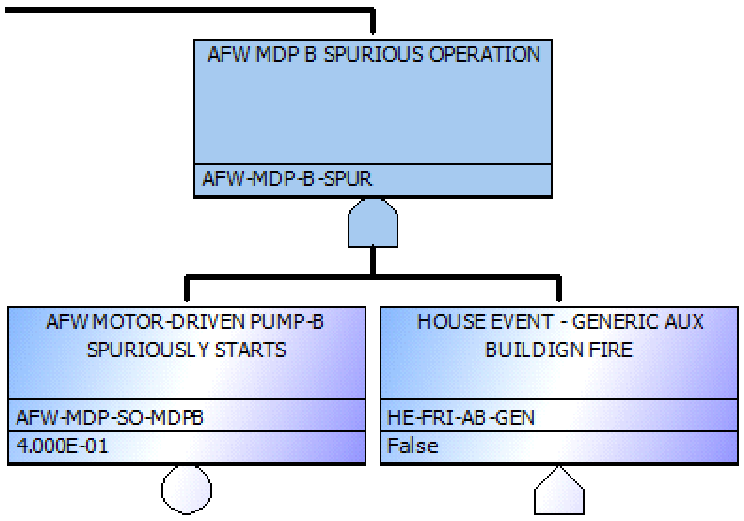
3.2 Fire Scenario Target Sets

The fire scenario target sets need to be added to the model. This part requires the majority of the work. First, the impact of the specific fire scenario on the plant must be identified. This impact can be anything from complete failure of component to spurious operation. The complete failure is straightforward within SAPHIRE; however, the spurious operation requires additional work. The complete failure will be discussed first, followed by the more complex modeling of spurious operation or adjustments to operator actions.

Complete failures are handled with SAPHIRE by developing fire scenario flag sets. A flag set is designed to tell SAPHIRE how to evaluate the accident sequences. Flag sets are assigned to each accident sequence prior to generating the cut sets. Within the flag set, the component is guaranteed to fail and is set to "TRUE." This tells SAPHIRE that, prior to cut set generation, the accident sequence logic can be pruned conditioned on the failure of the identified component. A flag set is developed for each fire scenario and contains the components (i.e., target sets) that are guaranteed to fail.

The more complex modeling accounts for spurious operation of components, but requires fault tree logic changes in order to apply alternate failure probabilities. This requires actual fault tree logic changes. For example, if a fire scenario can cause the spurious starting of an auxiliary feedwater (AFW) motor-driven pump with the spurious failure probability of 0.4 (just a generic probability for illustration purposes). This failure probability needs to be added as an additional failure mode of the AFW motor-driven pump. To perform this logic adjustment, a new basic event is added to the fault tree logic which is ANDed with the fire-specific house event. This logic is shown in **Error! Reference source not found.**.

Figure 4: Example Spurious Operation Modeling


AFW MDP B SPURIOUS OPERATION
AFW-MDP-B-SPUR
AFW MOTOR-DRIVEN PUMP-B SPURIOUSLY STARTS
AFW-MDP-SO-MDPB
4.000E-01
HOUSE EVENT - GENERIC AUX BUILDIGN FIRE
HE-FRI-AB-GEN
False

The house event shown in Figure 4 will be added to the flag set for this specific fire scenario. In the flag set, the house event, HE-FRI-AB-GEN, will be set to "TRUE," allowing the spurious start of the AFW motor-driven pump to be generated as a potential failure of this particular pump.

If there are operator action adjustments required for the specific fire scenario, the same type of modeling is required. A new operator action basic event is added to the logic and ANDed with the house event for that specific fire scenario. The fire-specific house event is then part of the flag set that gets assigned to the fire scenario accident sequences. The setting of the house event will then allow the new operator action to be generated into the final fire scenario cut sets.

The adjustment to all potential fire scenario target sets is required to be included into the fire scenario specific flag set, both complete failure or spurious operation. The flag set is then assigned to the fire scenario accident sequences using a link event tree rule. This linkage rule allows for the flag set to be assigned to the fire scenario accident sequences. The following rule is an example event tree linkage rule that is necessary to properly analyze fire scenarios:

```
if always then
  eventree(fire scenario) = FLAG(fire scenario flag);
  eventree(fire scenario) = ENDSTATE(CD-FIRE);
endif
```

The rule first defines the search criteria. In this case, all of the fire scenario accident sequences meet the criteria (i.e., always). The second line will now append the defined flag set to all accident sequences. As discussed, this flag set will fail the fire target sets and make any other defined logic changes. The third line will have the internal event accident sequences end state change from its nominal core damage (CD) to the user-defined fire CD. This allows for all fire scenarios to be grouped into a single end state. Now that the linkage rule has been developed and applied to the fire scenario accident sequences, these sequences are ready to be analyzed.

The illustration above is a high-level view of the process. The actual development is more complex and time consuming. The last part of the fire scenario modeling is the quantification and cut set generation. This process is discussed in the next section.

3.3 Quantification of Fire Scenarios

Quantification is the last part of fire modeling. Once all the fire scenarios have been identified, the event trees developed, flag sets created and linked to the fire scenario accident sequences, minimal cut sets can be generated. Minimal cut sets represent those combinations that are both sufficient and necessary to lead to the failure of accident sequence. These minimal cut sets are then gathered into the single fire CD end state. This end state gathering process is important due to the potential of similar minimal cut sets showing up in different accident sequences. The reason similar minimal cut sets would show up in different accident sequences is due to the multiple failure sequences and not handling the success terms through the accident sequences.

The CD fire end state can now be quantified to obtain the overall fire core damage frequency (CDF). The quantification process of these cut sets can utilize different options. The default option utilized by most PRA software programs is the minimal cut set upper-bound approximation, 1-Product(1-cut set(i)) i = 1 to n. This grouped end state of fire scenario cut sets can then be separated into the individual fire scenarios to obtain the individual fire scenario CDF and its percent contribution to the overall fire CDF. Comparisons of the fire CDF to the internal events CDF can also be obtained.

4. SAPHIRE SCRIPTING MODIFICATION

SAPHIRE has a robust interactive graphical user interface to allow users to add basic events, fault trees, event trees, and other information to create and update their PRA models. To automatically test many of the functions that support the user interface, a collection of scripting or macro functions have been created. These macros can perform many of the functions required in a repeatable manner to promote the setup and evaluation of a model with various model changes. SAPHIRE can load various types of data through its Models and Results Database (MAR-D) Input capability. A brief overview of the MAR-D Input capability and macros are presented below.

4.1 MAR-D Input

MAR-D provides an interface to load or extract data files that define the PRA database. The files are in a "flat-file" or ASCII file format. These files can be constructed or modified manually or programmatically, and then used by the macro script to add to or modify a model. For this project the MAR-D files are constructed by FRI3D, described in Section 5. The MAR-D file text format and field descriptions are provided in SAPHIRE reference material [12] (NUREG/CR-7039, Volume 7).

Typical uses of MAR-D include:

- Transfer PRA information between databases.

- Extracting MAR-D files from one SAPHIRE project and loading them (via MAR-D) into another SAPHIRE project. (The SAPHIRE project may be a new one or a previously existing project.)
- Import other PRA code information.
- Formatting the model information from another PRA code to use MAR-D file formats and creating a SAPHIRE project by loading the files via MAR-D.
- Edit PRA files using a text editor.
- Extracting MAR-D files from a SAPHIRE project, editing the files to make changes to the model or model descriptions, and loading those files (via MAR-D) back into the SAPHIRE project.
- Archiving PRA files.
- Saving the MAR-D files for long-term storage in a text format rather than the native binary SAPHIRE format.

Details on the MAR-D formatting rules are also found in SAPHIRE reference material [12] (NUREG/CR-7039, Volume 7). Several MAR-D input files are used to update the selected SAPHIRE project. There are several different file types, each one specifying data for different items in the SAPHRIE model. For example, Event Tree Names/Descriptions can be added with an *.ETD file. A list of the file types and what they specify in the SAPHIRE model can also be found in the SAPHIRE reference material [12].

4.2 Macros

Macros are scripts that can run automated SAPHIRE 8 routines. Macros can be imported, exported, and run through the user interface, or they can be placed on the command line as SAPHIRE is started. Various functions and reports can be automated using these macros. The macros are written in a language very similar to the XML language which defines classes, verbs, and parameters.

Classes correspond to PRA data objects like projects, fault trees, basic events, event trees, change sets, and end states. Verbs are functions that can be performed on the various data objects. Examples of verbs would be cut set generation for event trees or fault trees, event trees linking, or uncertainty analysis on generated cut sets. Parameters are used to shape the functions being performed by defining a truncation value, report name, or the sample size of an uncertainty analysis. There are over 250 keywords that define these classes, verbs, and parameters.

Below is a collection of keywords, a macro, that tells SAPHIRE to select an event tree and link it to create sequences.

```
<event tree>                    Opens the class of PRA object (event tree) we are going to
                                work with.
 <unmark></unmark>              Verb to make sure no event tree is marked.
 <mark mask>LOSP</mark mask>    Verb to select or mark the event tree (LOSP).
 <unlink></unlink>              Verb to remove all sequences of this event tree.
 <link></link>                  Verb to link the marked event tree.
</event tree>                   Close the class of PRA object we worked with.
```

It will then solve those sequences with a designated truncation value and then produce a summary report showing the overall answer and number of cut sets for each sequence.

```
<sequence>                                          Opens the class of PRA object (event tree)
                                                    we are going to work with.
 <unmark></unmark>                                  Verb to make sure no event tree is marked.
 <include>                                          Super Verb for explicit marking
  <mark event tree mask>LOSP</mark event tree mask> Verb to select the LOSP event tree
  <mask operation>and</mask operation>              Verb with parameter to define marking
```

<mark sequence mask>*</mark sequence mask>	Verb to select all sequences selected event tree
<mask operation>and</mask operation>	Verb with parameter to define marking
<mark logic fault tree>*</mark logic fault tree>	Verb to select any sequences using the marked defined fault trees
</include>	Close of Super Verb for explicit marking
<solve>	Open of Verb to solve for cut sets and apply post processing rules
<with update></with update>	Parameter to indicate cut set update should be done
<truncation>0.00</truncation>	Parameter to indicate the truncation to be applied
</solve>	Close of Verb
<report>	Open of Verb to perform a report
<type>use title</type>	Parameter defining the report type – use title
<title>Sequence Results Compare</title>	Parameter defining the report title
<file name>current_vs_base.html</file name>	Parameter defining the name of the report
<report format>html</report format>	Parameter defining the output type (PDF, HTML, etc)
</report>	Close of Report Verb
</sequence>	Close the class of PRA object we worked with.

5. AUTOMATED FIRE PRA SCENARIOS

The SAPHIRE solve module uses both the MAR-D Input and Macros features described in Section 5 that were used to add scenarios to SAPHIRE in the format outlined in Section 4. This module took approximately 2 weeks of development work and testing. The following sections describe how this was done.

5.1 FRI3D Scenarios to MAR-D

To construct proper MAR-D files, SAPHIRE uses a set of MAR-D templates for the following input file extensions (.ETA, .ETD, .ETL, .ETR, .CSI, .CSD, .CSA, .ESD, .FTD, .FTL, .BEI, .BEA, .EDG, and .EGI) described in Section 4.1. Each of these has a key marker for text replacements.

- Event Trees – For each scenario the user selected to solve, a new event tree is defined using the .ETA, .ETD, .ETL, .ESD files.
- Initiating Events – An initiating event is created for each scenario with the value = (non-suppression probability *severity factor * ignition frequency) using the (.BEI and .BEA) files.
- Linkage Rules – A linkage rule is also created for each scenario, applying the correct flag set to fail the scenario events given the fire initiator. This is done in the (.ETR) file.
- Event Tree Group – A fire event tree group is added with each of the fire event trees through the (.EGD and .EGI) files.
- Flag Sets – A flag set is created for each scenario containing all the basic events that fail for that scenario. These are using the (.CSI, .CSD, and .CSA) files.

5.2 FRI3D Running SAPHIRE

Once the MAR-D files are created, they need to be applied to a base SAPHIRE model and then the scenarios solved. This is done using SAPHIRE's macro script. This script first specifies the MAR-D files to be loaded into the model using a "Scenario" block. Next, the fire event trees are linked using a mask of "FRI-*" and set to solve. Finally, the fire end state is marked and set to solve.

Figure 5: Snipit of the SAPHIRE Macro Script Generated from FRI3D

```
<scenario>
  <start>
    <name>%P-02</name>
    <description>Load Files</description>
  </start>
</scenario>
<comment> ***********************LOAD TEMPLATE DATA
********************************</comment>
<basic event>
  <load>
    <type>bea</type>
    <file name>%%ProjPath%%FRI3D_Subs\FRI3D.bea</file name>
...
<scenario>
  <end>
  </end>
</scenario>
<comment> link and solve all the fire event trees then gather </comment>
<event tree>
  <unmark></unmark>
  <mark mask>FRI-*</mark mask>
  <unlink></unlink>
  <link></link>
  <solve>
...
<end state>
  <unmark></unmark>
  <mark name>CD-FRI</mark name>
  <solve>
    <gather method>by sequence</gather method>
...
```

The user assigns the location of the SAPHIRE application and the base model to be used. When they solve the scenarios, the base model and MAR-D files are copied to a temporary directory, and SAPHIRE is started from FRI3D with the macro file and base model passed in as parameters.

6. GENERIC SPAR MODEL EXAMPLE

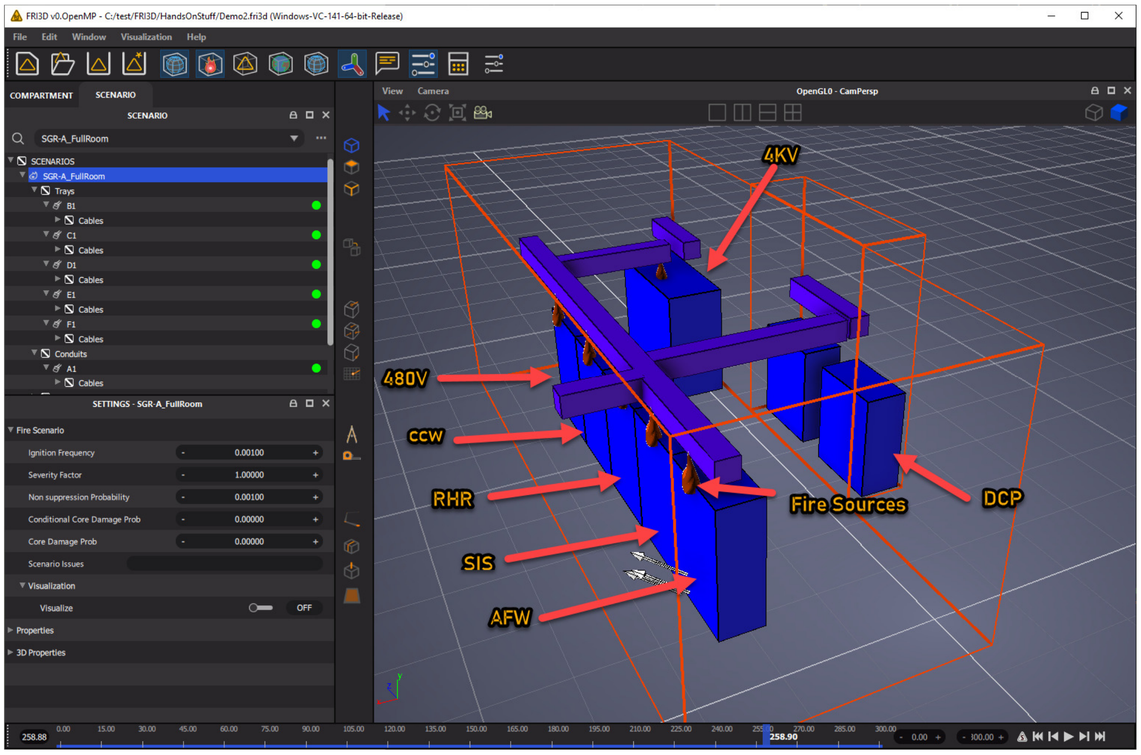
The SAPHIRE solving feature was used in a pilot evaluation on INL's Advanced Test Reactor PRA model. However, for this demonstration, the generic pressurized water reactor (PWR) SAPHIRE model was used.

6.1 FRI3D Model

First, a hypothetical layout for a switch gear room was made. A basic fire logic mapping was also constructed to import into a new FRI3D model to begin modeling. The switch gear room consists of power cabinets for the A train of the following SAPHIRE components: 4KV, 480V, CCW, RHR, SIS, AFW, and DCP[*]. Raceways connect the different cabinets with a hypothetical configuration of cables running from 4KV to the other power cabinets, and power cables from the cabinets to their components. **Error! Reference source not found.**, shows the 3D model constructed in FRI3D with the cabinets labeled.

Figure 6: Hypothetical Switch Gear Room Modeled in FRI3D for the Generic PWR SAPHIRE Model

[*] See component naming descriptions at the end of the document.


FRI3D v0.OpenMP - C:/test/FRI3D/HandsOnStuff/Demo2.fri3d (Windows-VC-141-64-bit-Release)
File Edit Window Visualization Help
COMPARTMENT SCENARIO
SCENARIO
View Camera
OpenGL0 - CamPersp
4KV
480V
CCW
RHR
SIS
AFW
Fire Sources
DCP

SGR-A_FullRoom

- SCENARIOS
  - SGR-A_FullRoom
    - Trays
      - B1
        - Cables
      - C1
        - Cables
      - D1
        - Cables
      - E1
        - Cables
      - F1
        - Cables
    - Conduits
      - A1
        - Cables

SETTINGS - SGR-A_FullRoom

Fire Scenario

Ignition Frequency	0.00100
Severity Factor	1.00000
Non suppression Probability	0.00100
Conditional Core Damage Prob	0.00000
Core Damage Prob	0.00000
Scenario Issues	

Visualization

Visualize OFF

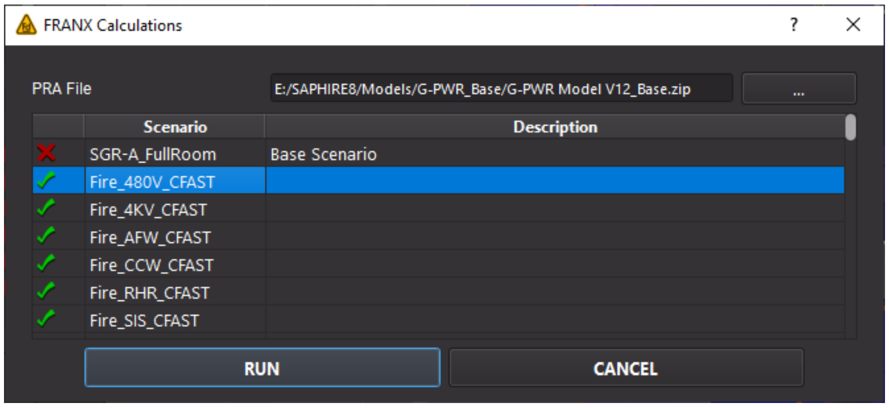
Properties

3D Properties

6.2 Scenarios

A fire source for each of the cabinets was added using a medium T-Squared HRR curve. Scenarios were auto-generated using CFAST results, and the fire logic mapping for determining basic events. After selecting the calculate button, the base SAPHIRE model was selected along with the scenarios to solve. Only the selected scenarios were used to generate the MAR-D files and added to the base model.

Figure 7: Scenarios Generated to be Sent to SAPHIRE for a Conditional CDF Calculation



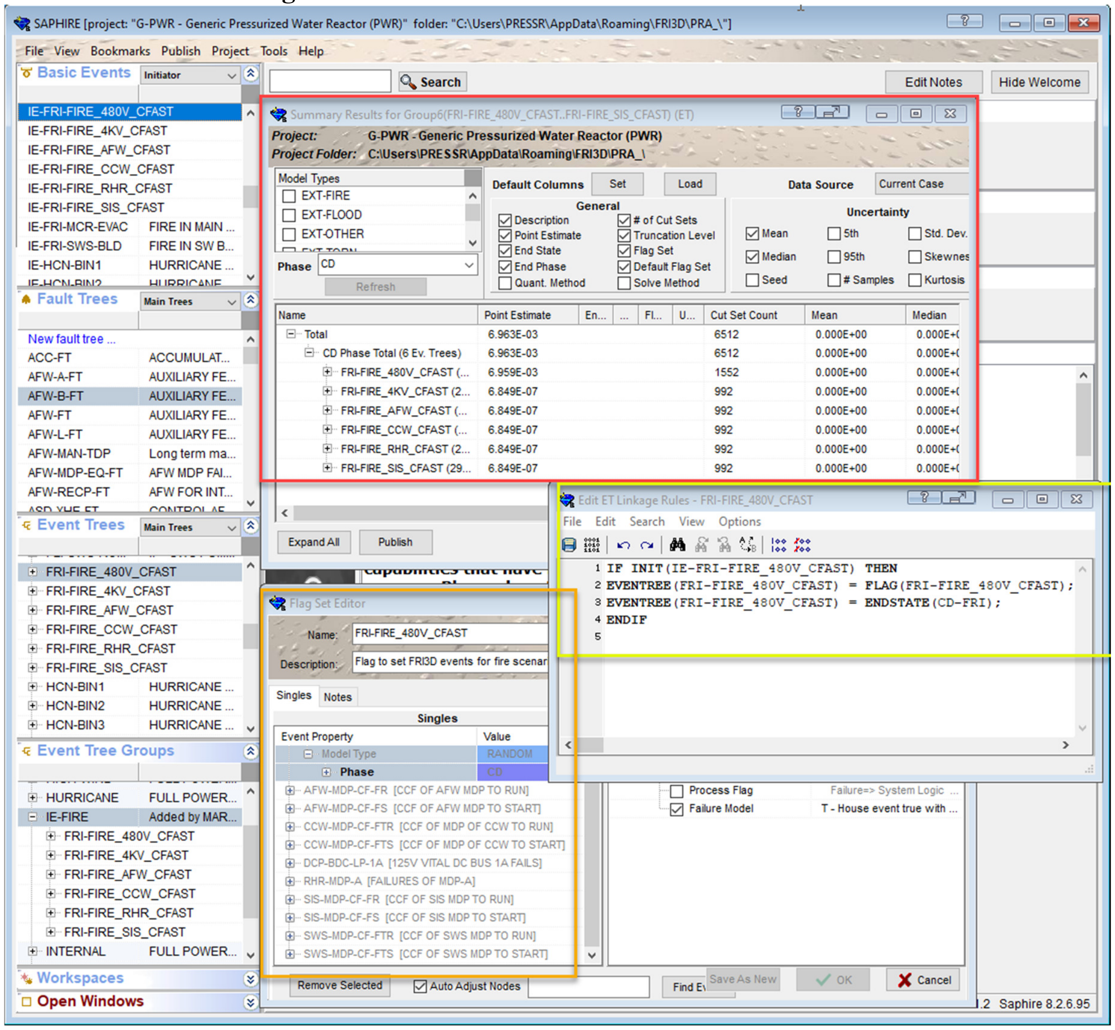
PRA File E:/SAPHIRE8/Models/G-PWR_Base/G-PWR Model V12_Base.zip

	Scenario	Description
✗	SGR-A_FullRoom	Base Scenario
✓	Fire_480V_CFAST	
✓	Fire_4KV_CFAST	
✓	Fire_AFW_CFAST	
✓	Fire_CCW_CFAST	
✓	Fire_RHR_CFAST	
✓	Fire_SIS_CFAST	

6.3 SAPHIRE Model

The SAPHIRE model opens and starts running the macro generated from FRI3D which imports the changes from the MAR-D files and solves the fire event trees. The resulting SAPHIRE model is shown in **Error! Reference source not found.**. The new fire event trees, initiating events, and the event tree group created can be seen on the left-hand-side lists. An example of the flag sets generated is highlighted in orange, with each basic event being set to a "House event true." The results for all the fire scenarios can be seen by right clicking on the fire event tree group and selecting "View Summary Results." This is highlighted in red. An example of the event tree linkage rules is shown highlighted in yellow.

Figure 8. SAPHIRE Model Generated from FRI3D


SAPHIRE [project: "G-PWR - Generic Pressurized Water Reactor (PWR)" folder: "C:\Users\PRESSR\AppData\Roaming\FRI3D\PRA_\"]

Summary Results for Group6(FRI-FIRE_480V_CFAST..FRI-FIRE_SIS_CFAST) (ET)

Project: G-PWR - Generic Pressurized Water Reactor (PWR)
Project Folder: C:\Users\PRESSR\AppData\Roaming\FRI3D\PRA_\

Phase: CD

Name	Point Estimate	Cut Set Count	Mean	Median
Total	6.963E-03	6512	0.000E+00	0.000E+(
CD Phase Total (6 Ev. Trees)	6.963E-03	6512	0.000E+00	0.000E+(
FRI-FIRE_480V_CFAST (...	6.959E-03	1552	0.000E+00	0.000E+(
FRI-FIRE_4KV_CFAST (2...	6.849E-07	992	0.000E+00	0.000E+(
FRI-FIRE_AFW_CFAST (...	6.849E-07	992	0.000E+00	0.000E+(
FRI-FIRE_CCW_CFAST (...	6.849E-07	992	0.000E+00	0.000E+(
FRI-FIRE_RHR_CFAST (2...	6.849E-07	992	0.000E+00	0.000E+(
FRI-FIRE_SIS_CFAST (29...	6.849E-07	992	0.000E+00	0.000E+(

Edit ET Linkage Rules - FRI-FIRE_480V_CFAST

```
IF INIT(IE-FRI-FIRE_480V_CFAST) THEN
EVENTREE(FRI-FIRE_480V_CFAST) = FLAG(FRI-FIRE_480V_CFAST);
EVENTREE(FRI-FIRE_480V_CFAST) = ENDSTATE(CD-FRI);
ENDIF
```

Flag Set Editor

Name: FRI-FIRE_480V_CFAST
Description: Flag to set FRI3D events for fire scenar

Event Property	Value
Model Type	RANDOM
Phase	CD
AFW-MDP-CF-FR [CCF OF AFW MDP TO RUN]	
AFW-MDP-CF-FS [CCF OF AFW MDP TO START]	
CCW-MDP-CF-FTR [CCF OF MDP OF CCW TO RUN]	
CCW-MDP-CF-FTS [CCF OF MDP OF CCW TO START]	
DCP-BDC-LP-1A [125V VITAL DC BUS 1A FAILS]	
RHR-MDP-A [FAILURES OF MDP-A]	
SIS-MDP-CF-FR [CCF OF SIS MDP TO RUN]	
SIS-MDP-CF-FS [CCF OF SIS MDP TO START]	
SWS-MDP-CF-FTR [CCF OF SWS MDP TO RUN]	
SWS-MDP-CF-FTS [CCF OF SWS MDP TO START]	

Process Flag: Failure=> System Logic ...
Failure Model: T - House event true with ...

7. CONCLUSION

The methods for performing fire modeling in SAPHIRE for the SPAR models were successfully added to FRI3D. This was demonstrated with both an actual facilities SAPHIRE PRA model and with the generic PWR model, adding fire scenarios and solving for a conditional CDP. This work showed that it is relatively simple to add additional solving tools to the FRI3D software. Previous demonstration cases have shown how FRI3D can simplify and reduce many of the costs associated with fire modeling. Previous work has estimated a 50% reduction in scenario development time using FRI3D over traditional methods for new fire scenario development [13]. An additional 15–30 min. is typically needed to add each scenario to SAPHIRE, which would be reduced to seconds using FRI3D. The modular design of FRI3D made it easy to include SAPHIRE for quantification, and it can be easily adapted for facilities using other tools and methods.

Acknowledgments

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

References

[1] "EPRI/NRC-RES Fire PRA Methodology for Nuclear Power Facilities Volume 2: Detailed Methodology." EPRI 1011989, NUREG/CR-6850, Final Report. (September 2005). https://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr6850/v2/cr6850v2.pdf.

[2] "CFAST, Fire Growth and Smoke Transport Modeling." (May 2010, updated October 2019). https://www.nist.gov/el/fire-research-division-73300/product-services/consolidated-fire-and-smoke-transport-model-cfast.

[3] K. McGrattan, R. McDermott, C. Weinschenk, and G. Forney. "Fire Dynamics Simulator Users Guide, Sixth Edition, Special Publication (NIST SP)." National Institute of Standards and Technology, Gaithersburg, MD, (2013). [online], https://www.nist.gov/publications/fire-dynamics-simulator-users-guide-sixth-edition.

[4] "CAFTA, Computer Aided Fault Tree Analysis System, Version 5.4." (2009). [online], https://www.epri.com/research/products/000000000001015513.

[5] [online], https://saphire.inl.gov.

[6] S. Prescott, R. Christian, R. Sampath, and J. Biersdorf. "Fire Risk Investigation in 3D (FRI3D) Software and Process for Integrated Fire Modeling." INL/EXT-20-59506, Idaho National Laboratory, (2020).

[7] K. McGrattan, et al. "Cable Heat Release, Ignition, and Spread in Tray Installations During Fire (CHRISTFIRE) Phase 1: Horizontal Trays." NUREG/CR-7010 Vol. 2, U.S. Nuclear Regulatory Commission, (2012).

[8] K. McGrattan. "Thermally-Induced Electrical Failure (THIEF) Model." NUREG/CR-6931, Volume 3, (2008).

[9] U.S. NRC. "Refining and Characterizing Heat Release Rates from Electrical Enclosures During Fire (RACHELLE-FIRE) –Volume 2: Fire Modeling Guidance for Electrical Cabinets, Electric Motors, Indoor Dry Transformers, and the Main Control Board, Appendix A." NUREG-2178 Volume 2, EPRI 3002016052.

[10] U.S. NRC. "Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 8." NUREG/CR-7039, (March 2021). https://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr7039/index.html

[11] "Saphire Training." [online], https://saphire.inl.gov/#/training

[12] U.S. NRC. "Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE) Version 8: Data Loading." (June 2011). https://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr7039/v7/index.html

[13] S. Prescott, et al. 2021. "Industry Level Integrated Fire Modeling Using Fire Risk Investigation in 3D (FRI3D)." INL/EXT-21-604079, Idaho National Laboratory.

[14] S.Prescott, et al. "Visualization and Automation of Fire Modeling using Fire Risk Investigation in 3D (FRI3D)" 2021 International Topical Meeting on Probabilistic Safety Assessment and Analysis, pages 655-664

*Component Naming Descriptions

4KV – 4,000 volt main power cabinet feeding the plant, going to the system cabinets.
480V – General 480 volt supply
CCW – Component Cooling Water
RHR – Residual Heat Removal
SIS – Safety Injection System
AFW – Auxiliary Feedwater
DCP – DC Power