

# **A Framework to Expand and Advance Probabilistic Risk Assessment to Support Small Modular Reactors**

**Curtis Smith  
David Schwieder  
Robert Nourgaliev  
Cherie Phelan  
Diego Mandelli  
Kellie Kvarfordt  
Robert Youngblood**

**September 2012**

The INL is a U.S. Department of Energy National Laboratory  
operated by Battelle Energy Alliance



# **A Framework to Expand and Advance Probabilistic Risk Assessment to Support Small Modular Reactors**

**Curtis Smith  
David Schwieder  
Robert Nourgaliev  
Cherie Phelan  
Diego Mandelli  
Kellie Kvarfordt  
Robert Youngblood**

**September 2012**

**Idaho National Laboratory  
Idaho Falls, Idaho 83415**

**<http://www.inl.gov>**

**Prepared for the  
U.S. Department of Energy  
Office of Nuclear Energy  
Under DOE Idaho Operations Office  
Contract DE-AC07-05ID14517**

# **A Framework to Expand and Advance Probabilistic Risk Assessment to Support Small Modular Reactors**

---

Proposed framework to support the development of a state-of-the-art PRA to predict the safety, security, safeguards, and performance of SMR systems

Curtis Smith  
David Schwieder  
Robert Nourgaliev  
Cherie Phelan  
Diego Mandelli  
Kellie Kvarfordt  
Robert Youngblood

---

# Table of Contents

- 1 INTRODUCTION..... 1
- 1.1 Scope..... 1
- 1.2 Background ..... 3
- 1.3 Benefits of the Proposed Approach ..... 4
- 2 FRAMEWORK ..... 5
- 2.1 The General Approach..... 5
  - 2.1.1 Safety Margins ..... 5
  - 2.1.2 Safety Margin Example ..... 7
  - 2.1.3 Types of Analysis to be used for the Safety Case ..... 8
  - 2.1.4 Proposed High-Level Architecture for the Advanced PRA Approach ..... 10
  - 2.1.5 Analysis Techniques for Scenarios and Safety Margins..... 13
  - 2.1.6 Probabilistic Thinking..... 17
- 2.2 On the Need for Simulation..... 18
- 2.3 Physics-based Simulation ..... 19
- 2.4 On the Need for Emulators ..... 21
  - 2.4.1 Physical Modeling ..... 21
  - 2.4.2 Modeling and Analysis in the Licensing Process..... 21
  - 2.4.3 Evolution of Licensing Practice ..... 23
  - 2.4.4 Analysis is a Key to Licensing..... 26
- 2.5 Information Processing ..... 27
- 2.6 Details of the Proposed Software Architecture ..... 28
- 3 PROBABILISTIC CALCULATIONS..... 33
- 3.1 DYNAMIC SIMULATION MODEL..... 33
- 3.2 STATIC PRA MODEL ..... 34
- 4 MECHANISTIC CALCULATIONS ..... 36
- 4.1 Interaction with System Analysis Tools..... 36
  - 4.1.1 Platform Technologies and Requirements for SATs..... 36
  - 4.1.2 Main Concepts of the Mechanistic Implementation..... 39
  - 4.1.3 Predictor-Corrector Feedback..... 41
- 4.2 Notes on Advanced Simulation Tools..... 42
- 4.3 References..... 47
- 5 IMPLEMENTATION PLAN..... 49
- A APPENDIX A -- SIMULATION USING EMERALD..... 51

A-1	Discrete Event Simulator .....	51
	Simulation Object Model .....	51
A-2	Graphical User Interface.....	55
	Controls Tab .....	56
	Model Tab .....	56
	Functions Tab .....	58
	Variates Tab .....	59
	Outputs Tab .....	60
B	APPENDIX B -- SYSTEM REPRESENTATION .....	62
B-1	Object Information Library .....	62
B-2	Graphical SBD Editor .....	64
B-3	SBD to Fault Tree Routine .....	65
C	APPENDIX C -- DATA MINING .....	67
	Cluster Analysis.....	68
	Survey of Classification Algorithms .....	70
	Survey of Clustering Algorithms .....	71
	Machine Learning .....	73
	Impact on Advanced PRA.....	73
	Manifold Analysis .....	73
	Impact on Advanced PRA.....	74
	Anomaly Detection .....	74
	Impact on Advanced PRA.....	75
	Tools available .....	75

## 1.1 Scope

During the early development of nuclear power plants, researchers and engineers focused on many aspects of plant operation, two of which were getting the newly-found technology to work and minimizing the likelihood of perceived accidents through redundancy and diversity. As time, and our experience, has progressed, the realization of plant operational risk/reliability has entered into the design, operation, and regulation of these plants. But, to date, we have only dabbled at the surface of risk and reliability technologies. For the next generation of small modular reactors (SMRs), it is imperative that these technologies evolve into an accepted, encompassing, validated, and integral part of the plant in order to reduce costs and to demonstrate safe operation. Further, while it is presumed that safety margins are substantial for proposed SMR designs, the depiction and demonstration of these margins needs to be better understood in order to optimize the licensing process. Currently, a variety of stated/unstated substantiated/unsubstantiated assumptions have been made for SMR plant designs, including:

- Greater safety margins
- Smaller exclusions zone (the traditional zone is too large)
- Simpler emergency planning
- Reliance on passive safety systems
- Smaller and delayed source term
- Accident doses will be lower

These, and other, statements may be accurate, but they will need to be substantiated as part of a risk-informed approach. Consequently, INL is proposing an approach to expand and advance the state-of-the-practice in probabilistic risk assessment (PRA), specifically we will:

*Develop a framework for applying modern computational tools to create advanced risk-based methods for identifying design vulnerabilities in SMRs. This task will require the fusion of state-of-the-art PRA methods, advanced visualization methods and high-performance optimization methods within a flexible open source framework. Initial effort will be to define the conceptual framework and a draft implementation plan.*

It is the **purpose of this document** to support the development and planning for a framework for applying modern computational tools to create advanced risk-based methods for identifying design vulnerabilities in SMRs.

This document provides:

- The INL findings and recommendations for defining the conceptual framework
  - A draft implementation plan to carry out the development and demonstration of the framework
-

In order to support an optimized licensing process, we need a framework and process that will provide:

- Support for existing regulatory approaches
- Support for the development of methods and tools to predict safety margins
- Support for a demonstration of quantitative safety margins for specific technologies and designs

Ultimately, we will need an analysis approach to predict, via a safety case, the technical basis behind margins that impact performance measures such as safety (note that other performance measures such as economics are important, but are not discussed here). In addition, we need to tailor these methods and tools in order to use a graded approach both for short- versus long-term applications.

A key area of the SMR PRA strategy is the development of methodologies and tools that will be used to predict the safety, security, safeguards, performance, and deployment viability of SMR systems. The **goal of the SMR PRA framework development** will be to provide quantitative methods and tools and the associated analysis approach for assessing a variety of risks. These risks will be focused on SMR designs and operational strategies as they relate to an understanding of the technical basis behind safety and security characterization.

The development and implementation of SMR-focused safety assessment methods may require new analytic methods or an adaptation of traditional methods to the new design and operational features of SMRs. We will need to move beyond the current limitations of static, logic-based models in order to provide more integrated, scenario-based models based upon predictive modeling which are tied to factors that may cause failures or degradations. Ultimately, the development of SMR-specific safety models for margin determination will provide a safety case that describes potential accidents, design options including controls, and supports licensing activities by providing a technical basis for the safety envelope.

The SMR PRA framework will focus on the support for computational approaches and tools that will be used to conduct analysis of security and safety performance of the various SMR technologies. As a part of this approach, the use of PRA will be explored, where security- and safety-related scenarios will be described via probabilistic models with the intent to support risk-informed decision making. In this framework, the following areas will be considered:

- Representation of specific SMR design issues such as having collocated modules and passive safety features.
- Use of modern open-source or readily available analysis methods and software to support the probabilistic modeling.
- Emergency planning and management, including source term evaluation.
- Internal and external events resulting in impacts to safety.
- All-hazards considerations including the reactor core, storage/movement of spent fuel, and hazardous gases.
- Risks that may be present during low-power and shutdown conditions.
- Methods to support the identification of design vulnerabilities.
- Mechanistic and probabilistic data needs to support the modeling and tools development effort.

## 1.2 Background

It is readily acknowledged by many in nuclear technology fields that the quantitative risk and reliability aspect of nuclear power plant (NPP) operation is a vital part of future growth within the industry. Currently, this risk aspect is felt in the design and licensing of new NPPs and in the operation of current plants [e.g., the U.S. Nuclear Regulatory Commission's (NRC) Maintenance Rule, 10 CFR 50.65, and the Significance Determination Process (SDP)]. But, the current regime for quantitative risk and reliability assessment utilizes tools and techniques that are, in some cases, over thirty years old. Consequently, researchers in the risk and reliability sciences for NPPs have not been able to effectively utilize the advances in areas such as:

- Parallel and advanced computation techniques
- Dynamic simulation
- Aging-degradation of materials
- Embedded systems and advanced sensors
- Virtual environments
- Human cognition modeling
- Information technologies
- Parameter Data Analysis

The approach outlined in this document addresses the formulation and development of a risk/reliability platform for next generation SMR safety analysis. Since the task of measuring and managing risk of any complex, technological system is a multi-disciplinary endeavor; the objectives outlined for this project necessarily encompass a variety of sciences. As such, we describe various tools and techniques, ranging from computer science to mechanistic engineering calculations, that will be required as part of an integrated PRA approach.



### 1.3 Benefits of the Proposed Approach

In later sections of this document, we describe the key parts and details of an enhanced PRA approach and toolkit. However, first we summarized the benefits of the proposed approach as compared to existing PRA methods and tools.

Capability	Existing PRA Limitations	Benefits of the Enhanced PRA
<b>Simulation of Accident Scenarios</b>	Limited treatment of dynamic sequence behavior (some discretization used, but this complicates modeling)	Able to capture timing considerations that may affect the safety margin and plant physical phenomena
<b>Safety Margin Evaluation</b>	Margin is not determined, instead discrete end states are decided upon during the model development	Safety margins will be determined directly by coupling mechanistic calculations with probabilistic calculations
<b>Spatial Interactions</b>	Very limited treatment of spatial interactions, mainly in select flooding and fire models	Physics-based 3D environments will capture spatial interactions as part of accident scenarios
<b>Failure Cause Representation</b>	Traditionally, specific failure causes are rolled up into failure models such as fails-to-run or fails-to-start	A robust database of failure causes, mechanisms, and models will be plugged-into the component library such that analysts may pick-and-choose failure modes.
<b>Cloud-based Creation, Analysis, and Storage of Safety Models</b>	Traditionally, safety analysis has been performed by individual risk analysts (or a small team of analysts) with limited sharing and computational support	Multi-discipline, engineering focused teams will be able to share both models and computational resources in order to perform advanced analysis

In this section, we describe key elements of the overall advanced PRA approach, including:

- The general approach for the advanced PRA approach
- The need for simulation-focused modeling
- Physics-based simulation methods
- The need for emulators representing physical processes
- Information processing as a part of PRA

## 2.1 The General Approach

In order to enable probabilistic aspects of nuclear-based systems, we will be developing a variety of infrastructure methods/models based upon simulation. The INL has extensive capability in classical severe accident risk analysis, but for future applications, the proposed development is much superior to the static logic-based approaches used in existing accident risk analysis, in which gross simplifications and numerical approximations are necessary. Successfully tying probabilistic system simulation to system physics is a key facet of advanced PRA methods and will directly address problems such as highly time-dependent scenarios in SMR risk analysis, where probability is a key aspect of the scenario.

Note that the science and engineering communities are increasingly moving to more sophisticated deterministic models in order to better represent the complexities of “the real world.” As part of this movement, we find an increasing reliance on or need for probabilistic approaches, including the elicitation of information. The use of probability concepts is needed to support the use of mechanistic models in a probabilistic world. Capturing what, why, and how one knows something related to science and engineering is important to realizing the potential of complex nuclear systems.

### 2.1.1 Safety Margins

In general terms, a “margin” is characterized in one of two ways:

- A deterministic margin, defined by the ratio (or, alternatively, the difference) of an applied capacity (i.e., strength) to the load. For example, we test a pressure tank to failure where the tank design is rated for a pressure **C**, it failed at pressure **L**, thus the margin is  $(L - C)$  (safety margin) or  $L/C$  (safety factor).
- A probabilistic margin, defined by the probability that the load exceeds the capacity. For example, we model failure of a pressure tank where the tank design capacity is a distribution  $f(C)$ , its loading condition is a second distribution  $f(L)$ , the probabilistic margin would be represented by the expression  $\Pr[f(L) > f(C)]$ .

This second type of margin, probabilistic, is shown in Figure 1. This figure demonstrates why probabilistic concepts are needed when evaluating margins – simply knowing the difference in “averages” of load and capacity is not sufficient when managing risks.

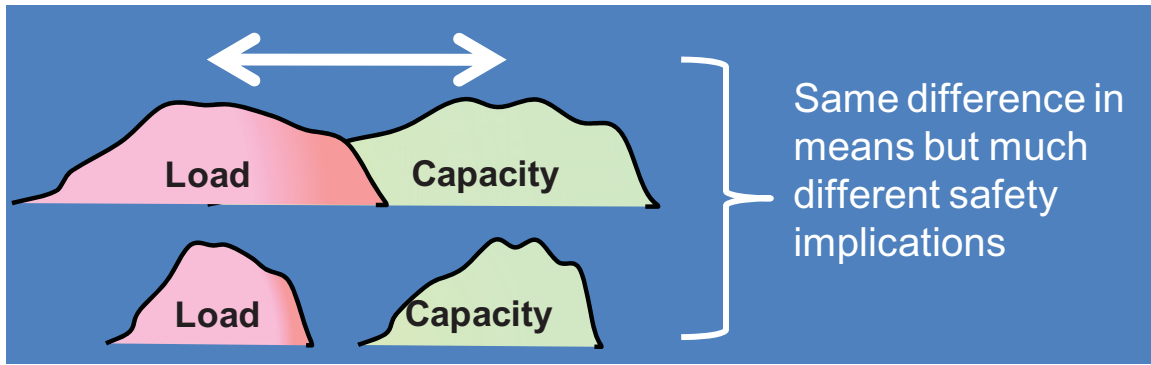


Figure 1. Representation of a probabilistic safety margin.

The safety margin focus we need for advanced PRA activities must consider realistic “load” and “capacity” implications for operating NPPs for a large variety of scenarios. For example, the notional diagram shown in Figure 2 illustrates that a safety impact, as represented by a load distribution, is a complex function that varies from one accident scenario to the next. However, the capacity part of the evaluation may not vary as much from one accident to the next because the safety capacity is determined by design elements such as fuel and material properties (which may be common across a spectrum of accidents).

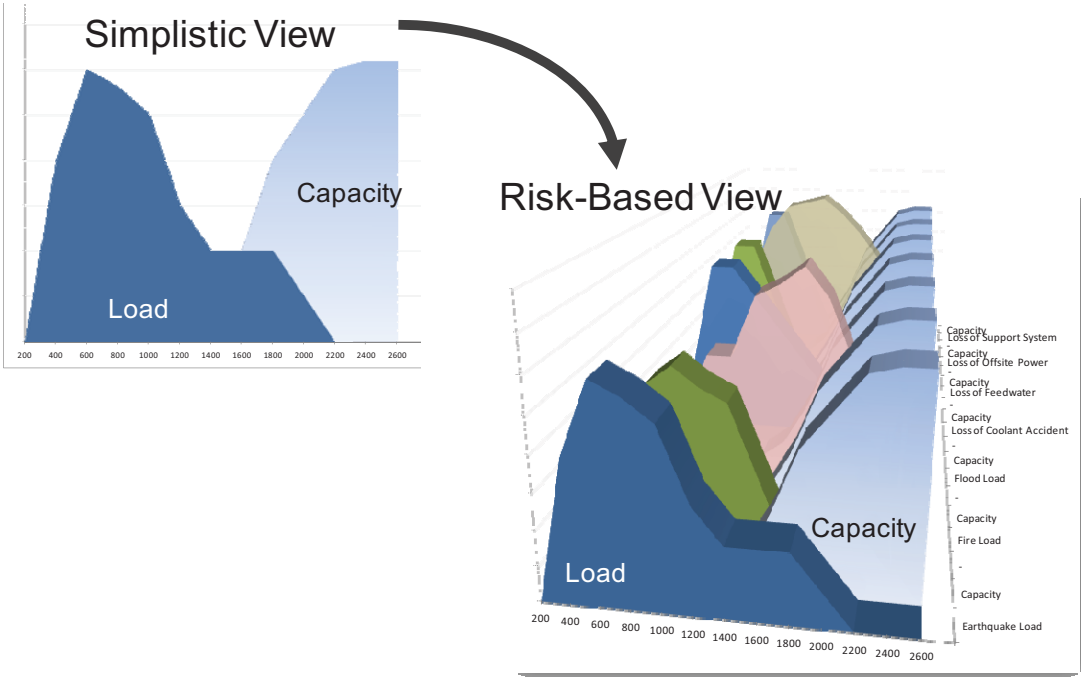


Figure 2. Family of load and capacity distributions representing different accident conditions.

## 2.1.2 Safety Margin Example

As an example of the type of results that are generated via probabilistic safety margins, we show a simple hypothetical example in Figure 3. For this example, we suppose that a NPP has two alternatives to consider: Alternative #1 – retain the existing, but aging, component as-is or Alternative #2 – replace the component with a new one. Using risk analysis methods and tools from the proposed framework, we run 30 simulations where this component plays a role in plant response

### Probabilistic Safety Margin

A numerical value quantifying the probability that a key safety metric (e.g., for an important process variable such as clad temperature) will be exceeded under specified accident scenario conditions.

under accident conditions. For each of the 30 simulations, we calculate the outcome of a selected safety metric – say peak clad temperature – and compare that against a capacity limit (assumed to be 2200 F). However, we have to run these simulations for both alternative cases (resulting in a total of 60 simulations). The results of these simulations are then used to determine the probabilistic margin:

Alternative #1:  $\Pr(\text{Load exceeds Capacity}) = 0.17$

Alternative #2:  $\Pr(\text{Load exceeds Capacity}) = 0.033$

If the safety margin characterization were the only decision factor, then Alternative #2 would be preferred (its safety characteristics are better). But, these insights are only part of the decision information that would be available to the decision maker, for example the costs and schedules related to the alternatives would also need to be considered.

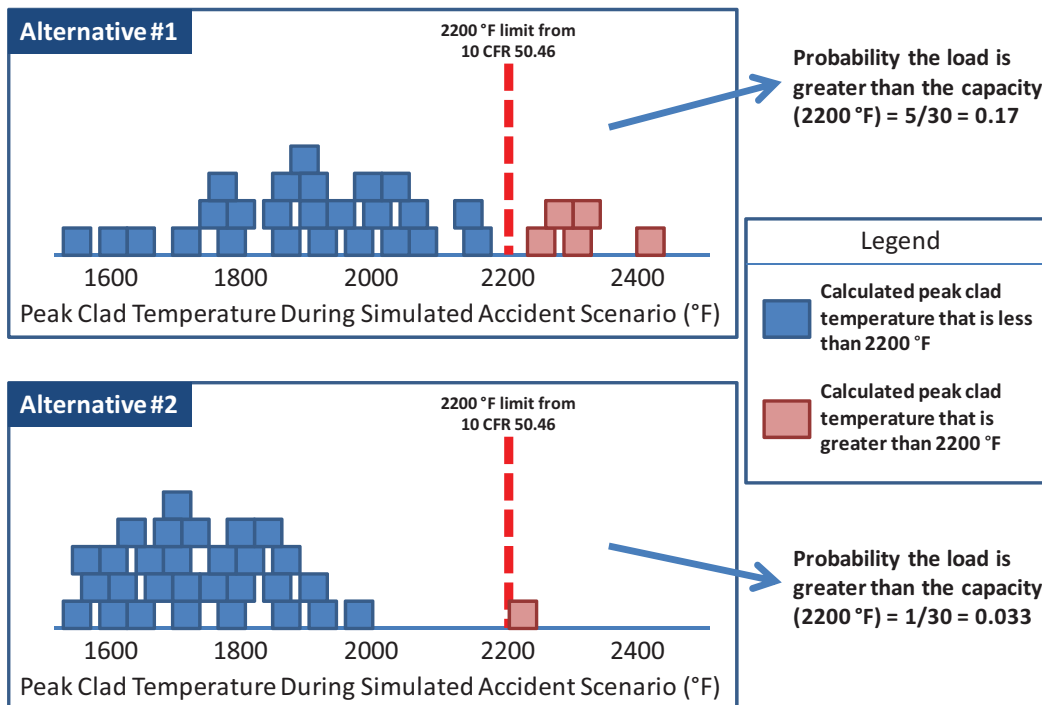


Figure 3. Safety margin example when evaluating changes to an SMR.

### 2.1.3 Types of Analysis to be used for the Safety Case

To better understand the approach to be used in the advanced PRA approach, we need to describe the two types of analysis used (see Table 1). Note that in actual applications, a blended approach is used where both types of analysis are used to support any one particular decision. For example, the approach could be either mostly probabilistic, mostly mechanistic, or both in nature.

Table 1. Types of analysis that are used in an advanced PRA approach

Types of Analysis Supporting the Safety Case	
PROBABILISTIC	MECHANISTIC
Pertaining to stochastic (non-deterministic) events, the outcome of which is described by a probability.	Pertaining to predictable events, the outcome of which is known with certainty if the inputs are known with certainty.
Probabilistic analysis uses models representing the randomness in the outcome of a process. Because probabilities are not observable quantities, we rely on models to estimate probabilities for certain specified outcomes.	Mechanistic analysis (also called “deterministic”) uses models to represents situations where the observable outcome will be known given a certain set of parameter values.
An example of a probabilistic model is the counting of k number of failures of an operating component in time t: $\text{Probability}(k=1) = \lambda e^{-\lambda t}$ .	An example of a mechanistic model is the one-dimensional transfer of heat (or heat flux) through a solid: $q = -k\partial T/\partial x$ .

The use of both types of analysis, probabilistic and mechanistic, is shown in Figure 4. Determining risk-based scenarios requires probabilistic considerations. Then, safety margin and uncertainty quantification rely on plant physics (e.g., thermal-hydraulics and reactor kinetics) coupled with probabilistic risk simulation. The coupling takes place through the interchange of physical parameters (e.g., pressures and temperatures) and operational or accident scenarios. These processes all support the safety case.

While definitions may vary in detail, the “safety case” essentially means the following:

*A structured argument, supported by a body of evidence that provides a compelling, comprehensible and valid case that a system is adequately safe for a given application in a given environment.*<sup>1</sup>

<sup>1</sup> Bishop, P. and R. Bloomfield, “A Methodology for Safety Case Development,” Safety-Critical Systems Symposium, Birmingham, UK. 1998.

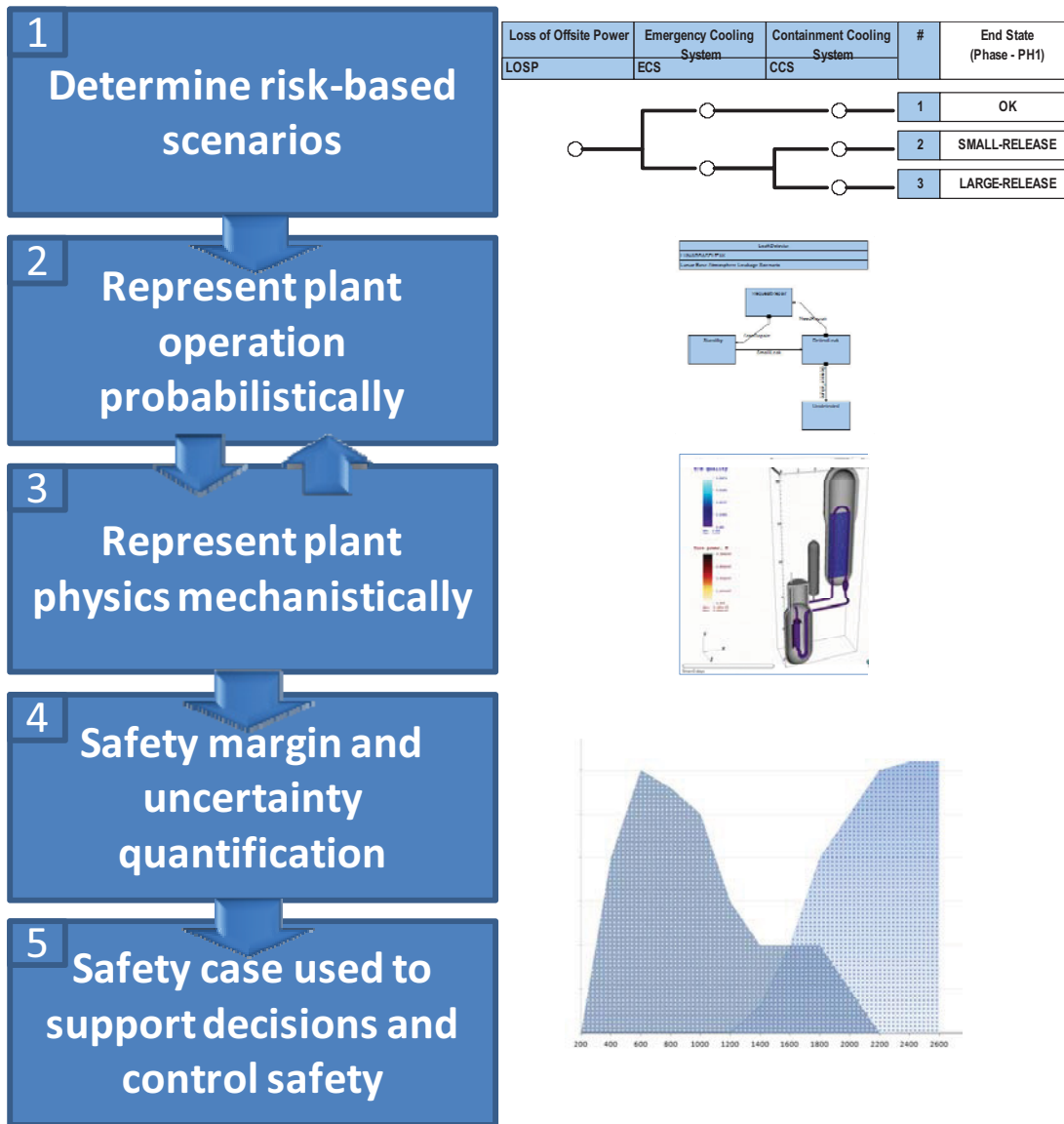


Figure 4. Attributes of the advanced PRA approach for supporting SMR decision-making.

The realization of a safety case will be an output of the SMR PRA framework. The safety-margin claims will do the following:

1. Make an explicit set of safety claims about the facility and SSCs
2. Produce evidence that supports the claims from #1 (e.g., representative operating history, redundancy in design, or results of analysis)
3. Provide a set of safety margin arguments that link the claims to the probabilistic and mechanistic evidence
4. Make clear the assumptions, models, data, and judgments underlying the arguments
5. Allow different viewpoints and levels of detail in a graded fashion for decision making.

The safety case should be regarded as having fundamental significance as opposed to being mere documentation of facility or SSC features. For practical purposes, “safety margin” is not observable in the way that many other operational attributes are (e.g., reactor core temperature). In decision-making regarding the facility or SSC, the safety case is, in practice, a proxy for the safety attribute. And, regardless of context, the formulation of a safety case is about developing a body of evidence and marshaling that evidence to inform a decision.

#### **2.1.4 Proposed High-Level Architecture for the Advanced PRA Approach**

The short description of the proposed approach is an Internet cloud-based PRA, where analysis modules (e.g., cut set solving of fault trees, simulation, mechanistic seismic calculations) are coupled to user-developed models in order to perform risk and vulnerability assessment. Since this approach uses Internet software advances, INL will be able to provide a scalable approach to PRA that is useable by a team of risk analysts – think of this as “the Google-Docs™” version of PRA.

The cloud-based PRA approach will allow for a well controlled analysis process, whereby "process" we imply the seamless integration of data + information + models + tools. Features of this new approach to PRA would allow analysts to:

- Obtain the most current data (e.g., failure rates, CCF factors, etc.) and models from a single spot, including providing support for data-related queries (e.g., "what is the failure rate for a particular component," "show me the seismic events in Model X")
- Use centralized analysis tools to (via a browser) perform analysis such as a vulnerability search, determine component importance, or perform “what if” analyses similar to those done at the U.S. Nuclear Regulatory Commission as part of the Significance Determination Process (SDP). These analyses could be shared with associates and regulators on an as-needed (and read-only) basis, thereby alleviating the need to send multiple files/models between team members. This would streamline analysis and understanding and make sure everyone has access to the right information at the right time.
- Augment the existing analysis (e.g., SAPHIRE calculations) with additional tools/techniques such as Bayesian model checks and simulation. For example, we could embed Bayesian analysis as a wizard-like interface to allow for defensible inference calculations that support the safety case.
- Provide an easily accessible, but secure, repository for these kinds of risk-informed analysis so future generations of analysts, plant designers, and regulators can learn about best practices.
- Expand the power of safety analysis over current PRA approaches by providing increased functionality, faster decision-response times, less downtime (the model and tools are available anywhere), improved working environments (no longer needing to “install” software – the best version is always the available version), and an integrated collaboration with analysts.
- Provide access support to supercomputing resources, thereby bypassing the limitation of having to develop and run analysis on a single desktop computer.

The proposed SMR PRA framework will have specific features, including:

- **Import** → Ties to various modeling and analysis packages in an open-framework manner.
- **Access** → Access the model on- or off-line and be able to store copies locally and in the cloud.
- **Collaborate** → Modeling and analysis that is conducted and reviewed in teams of scientists and engineers.
- **Integrate** → Integrate model, data, and information in order to have a holistic risk analysis and to minimize the number of technical models in use.
- **Organize** → Finding the right information at the right time (for analysts, users, and reviewers).
- **Synchronize** → In order to keep everyone on the same page and up-to-date.
- **Share** → Communication for decision makers, where key insights and uncertainties in the safety case are provided.
- **Secure** → Security of information is both a safety and business issue.
- **Analysis** → Generating evidence for the safety case.
- **Verify** → Demonstrating satisfaction of safety goals and identifying plan vulnerabilities so that controls can be implemented.

A high-level depiction of the major modules that would be required in the cloud-based PRA approach is shown in Figure 5. As shown in the figure, analysts and reviewers would access the analysis tools and PRA models by using an Internet browser. The other modules provide the modeling and analysis capabilities for the user.



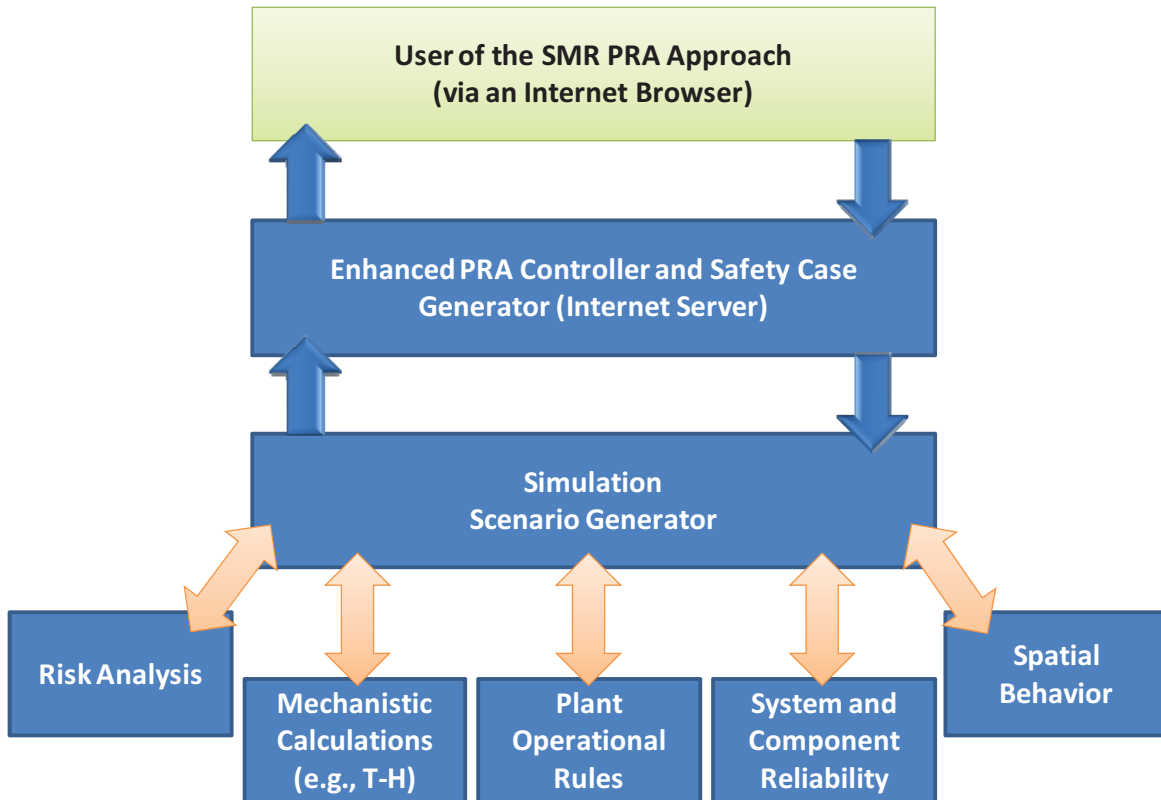


Figure 5. High-level architecture for the SMR advanced PRA framework.

The modules depicted above include:

- An overall PRA controller and module to produce (and store) the safety case. For each performance metric of consideration (e.g., peak clad temperature), this module will provide the time- and scenario-dependent results. These results (the “load”) will be contrasted to the capacity in order to determine safety margins. Engineering insights will be derived based upon the scenario and associated outcomes (both load and capacity) and are used to document the safety case.
- Simulation for scenario generation. Individual SSCs will be simulated to determine their operability (or not) over time. Coupled to the scenario generator (which is a probabilistic calculation) will be a variety of mechanistic calculations as needed for the scenario.
- Risk analysis including system-level failure models such as simulation, event trees, and fault trees.
- Mechanistic calculations will be used as needed to determine impacts to safety margins. For example, during a seismic event, load and capacity responses to the ground motion may be used to determine component operability.
- Plant operational rules, for example, “rules” including operator procedures, technical specifications, maintenance schedules.

- Knowledge of plant physical properties including systems, structures, and components (SSC). These models represent component failure models – failure causes and associated info (failures on starting, failures to run, failure rates, etc.).
- Spatial behavior will be used to determine interactions within (and possibly between) the power plant being evaluated. For example, if a fire causes a pipe rupture, the flow of water will be tracked to determine other possible failures in the scenario.
- Uncertainty quantification of both model and parameter sensitivities and uncertainties will be evaluated as part of the overall approach.

### **2.1.5 Analysis Techniques for Scenarios and Safety Margins**

One facet of the advanced PRA approach is to find vulnerabilities that affect margins. In general, a margins analysis approach for carrying out simulation-based studies of safety margin uses the following generic process steps:

1. Determine issue-specific, risk-based scenarios and accident timelines.
2. Represent plant operation probabilistically using the scenarios identified in Step 1.
3. Represent plant physics mechanistically.
4. Construct and quantify probabilistic load and capacity curves relating to safety to determine the safety margin.
5. Identify and characterize the factors and controls that determine safety margin within this issue to determine the safety case.

The general decomposition of what makes up an accident scenario is shown in Figure 6. We will use this representation to simulate overall plant behavior, possibly over long periods of time (years of operation). Since the state of “desired operation” is fairly well understood with regard to operational rules and mechanistic calculations [e.g., thermal-hydraulics (TH)], simulation calculations for this state tend to run very quickly.

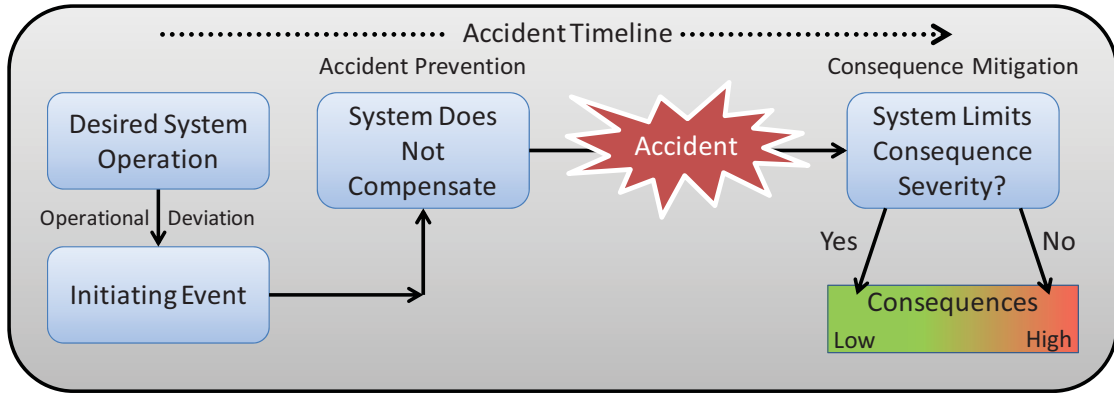


Figure 6. Accident scenario representation.

As we evaluate off-normal situations, the calculations that are required for plant simulation become more complex. For example, Figure 7 and Figure 8 show some of the types of probabilistic and mechanistic calculations (respectively) that would be required as part of the SMR PRA approach.

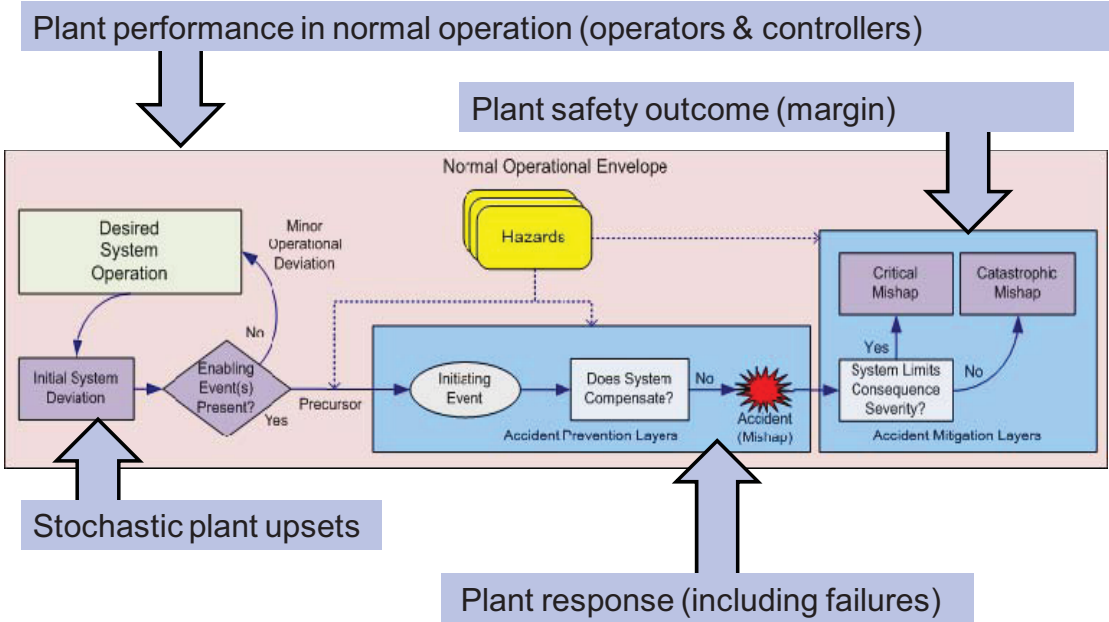


Figure 7. Characteristics of the accident scenario simulation for the probabilistic calculations.

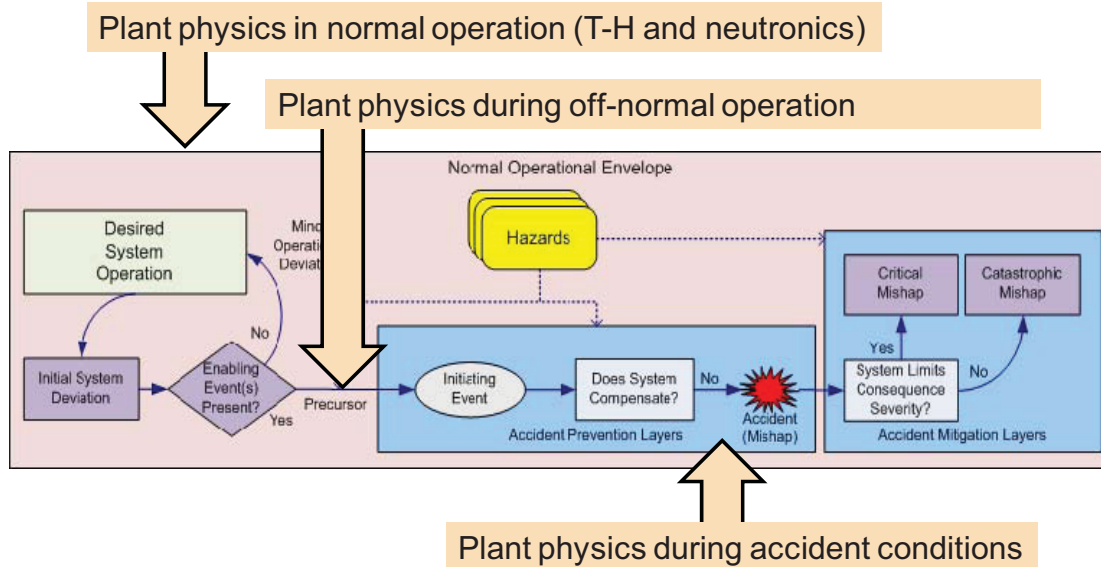


Figure 8. Characteristics of the accident scenario simulation for the mechanistic calculations.

One of the unique aspects of this proposed SMR PRA approach is in the use of existing PRA models such as event and fault trees. Later in this document we will describe the details of this aspect, but an example of the types of safety margin calculation we will perform for PRA is shown in Figure 9. In this figure, we show that traditional accident scenario models (in this case, an event tree) can inform the analysis as to the degree of margin for the accident scenarios described in the accident model. However, evaluating safety margins will potentially require an evaluation of all scenarios in the accident model, including those that are typically ignored for classical PRA approaches. Note that in Figure 8, the color-coded metric labeled PSMC indicates a Probabilistic Safety Margin Characterization, where a large margin is reflected via a large probability (i.e., close to 1 and shaded green) and a small (or nonexistent) margin has a low probability (i.e., close to 0 and shaded red).

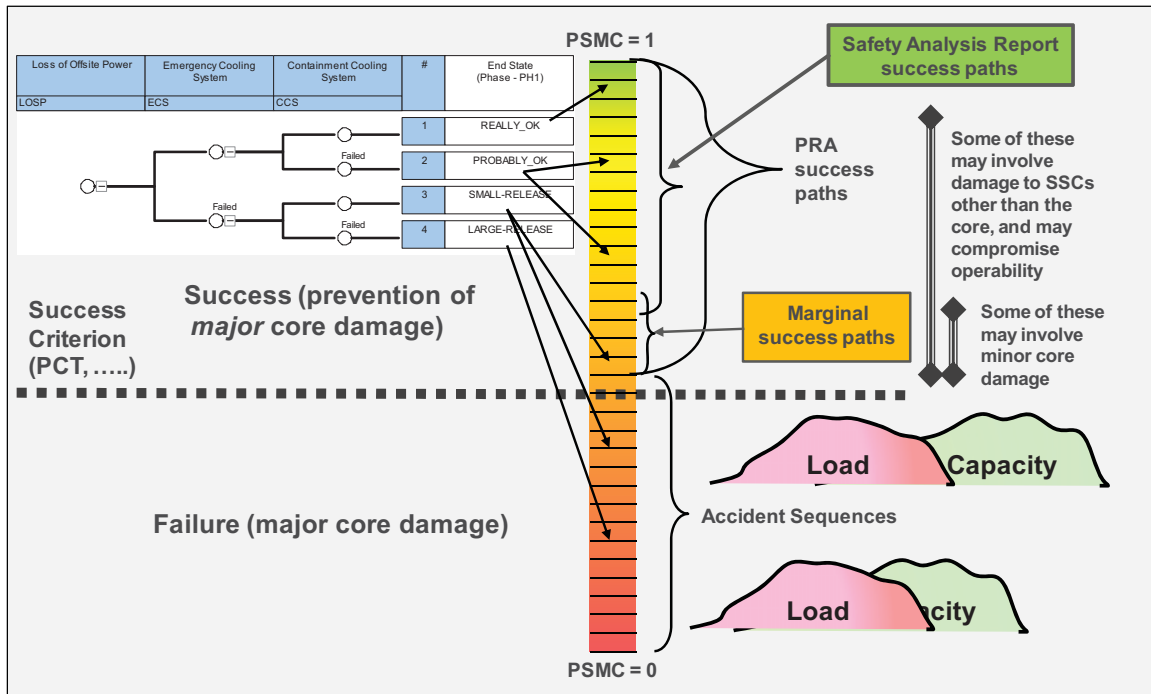


Figure 9. Safety margin evaluation building upon classical PRA models.

As we simulate these accident scenarios depicted above, we will realize the need to simulate a large number of scenarios. Consequently, later in this report we also describe the required computational power – note that this computational power (while modest by “supercomputer” standards) will be able to be shared by many users since the cloud aspect of the PRA tool allows the computations to be centralized.

Further, if we focus on the simulation details for each iteration, we see that we have unique information for the load on the plant SSCs and their capacity to respond to these impacts. Each individual scenario calculation makes up the totality of the load and capacity curves (providing the safety margin) as illustrated in Figure 10. For each of these scenarios, we will have information as to operation (or not) of individual SSCs, mechanistic evaluations (such as thermal-hydraulics), timing information, and estimates of the plant observables such as pressures, temperatures, and flow rates.

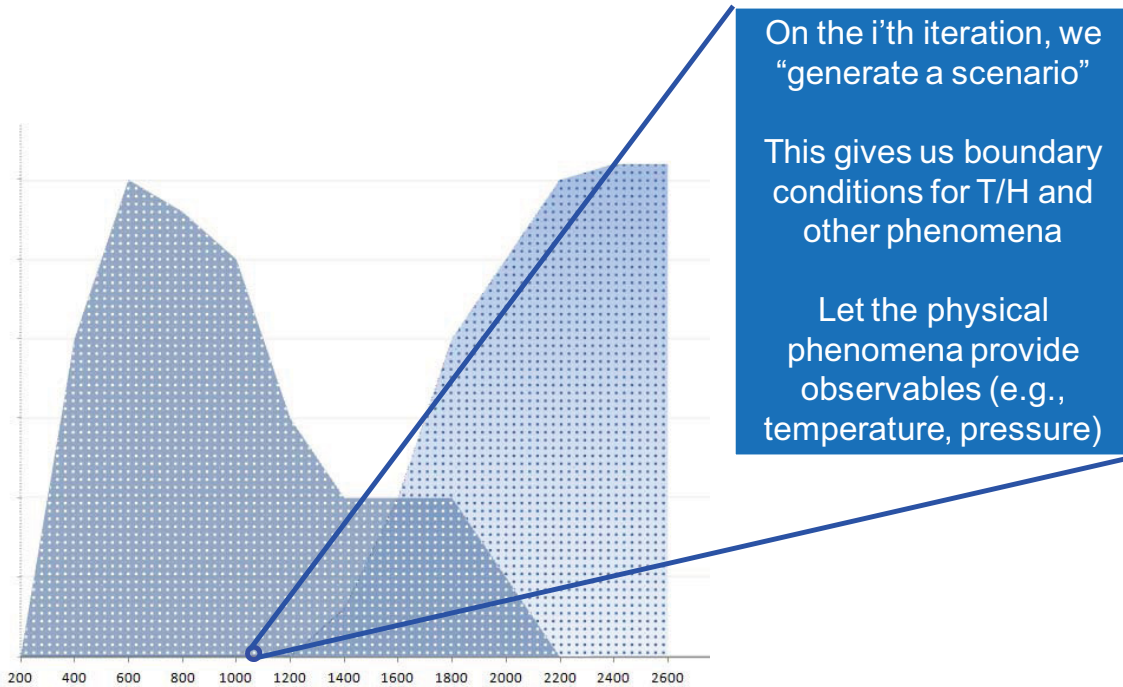


Figure 10. Example of one simulated accident scenario making up part of the safety margin.

### 2.1.6 Probabilistic Thinking

Underlying the entire SMR PRA framework is the notion of “probabilistic thinking,” specifically Bayesian probability concepts. While we have already used terms such as “data” and “information” we have not formally defined these, until now:

**Data**                 Distinct observed (e.g., measured) values of a physical process. Data may be factual or not, for example they may be subject to uncertainties, such as imprecision in measurement, truncation, and interpretation errors.

**Information**        The result of evaluating, processing, or organizing data/information in a way that adds to knowledge.

Examples of data include the **number** of failures during system testing, the **times** at which a component has failed and been repaired, and the **time** it takes until a heating element fails. In these examples, the measured or observed item is bolded to emphasize that data are observable. Note, however, that information is not necessarily observable; only the *subset* of information that is called data is observable. The availability of data/information, like other types of resources, is crucial to analysis and decision-making.

To evaluate or manipulate data, we must have a "model of the world" (or simply “model”) that allows us to translate real-world observables into information. Within this model of the world, there are two fundamental types of model abstractions, probabilistic and mechanistic, as we have already described.

The term “probabilistic” implies an inherent “randomness” in the outcome of a process. For example, flipping a coin<sup>2</sup> is modeled as an aleatory process, as is rolling a die. When flipping a coin, the “random” but observable data are the outcomes of the coin flip, that is, heads or tails. Note that since **probabilities** are **not** observable quantities, we do not have a model of the world directly for probabilities. Instead, we rely on probabilistic models (e.g., a Bernoulli<sup>3</sup> model) to predict probabilities for observable outcomes (e.g., two heads out of three tosses of the coin).

These kinds of probabilistic and mechanistic models will be built into the “component library” that is available for the SMR PRA application. These SMR models will be built upon existing models (e.g., failure rate models, common cause failures, aging models) as part of the cloud-based architecture, thereby ensuring that all analysts have access to the most comprehensive and applicable models for their PRA. Further, we have available advanced Bayesian inference analysis tools such as OpenBUGS that can be used to incorporate data into models using a probabilistic approach.

## 2.2 On the Need for Simulation

Since the SMR PRA approach is building upon existing PRA models such as fault trees, one question that may arise is why do we need to simulate failures and accident scenarios? The short answer is that we need to focus on safety margins, and in today’s modern risk-informed analysis and operational environments, having accurate answers at the appropriate time is critical to having a defensibly safety case. Consequently, it is necessary for the SMR PRA approach to develop and use enhanced capabilities targeting accuracy and timeliness. The primary way to satisfy these targets is to invest in new methods and tools related to simulation.

Simulation is a general modeling approach that can be used many areas. However, simulation is more than just a methodology – using simulation approaches forces one think holistically about complex system representations. Rather than focusing on how one can reduce a problem into a solution using existing approximate tools (like a fault tree), the evaluator is unencumbered and can, instead, focus on the problem (rather than the solution method). Consequently, simulation approaches can frequently provide relevant system insights that are simply not possible using older methods.

Further, simulation may be used to represent either mechanistic or probabilistic processes. For example, the analysis modules will include simulation to represent a variety of deterministic processes (e.g., fire environments, material damage, releases of material) during upset conditions. And, operability issues including reliability, risk, availability, maintainability, etc., have been evaluated using simulation in a variety of analysis-intensive private sector and government industries.

A common idiom is the phrase “the devil is in the details.” While a cultural reference, this phrase is relevant when discussing capabilities targeting accuracy and timeliness. The conventional PRA approach (e.g., static logic-based models) is considered acceptable for answering select types of questions in select types of systems under select types of conditions. However, as industries such

---

<sup>2</sup> Flipping a coin is mechanistic in principle, but the solution of the “coin-flipping dynamics” model, including knowledge of the relevant boundary conditions, is too difficult to determine or use in practice. Hence, we abstract the flipping process via a probabilistic model of the world.

<sup>3</sup> A Bernoulli trial is an experiment outcome that can be assigned to one of two possible states (e.g., success/failure, heads/tails, yes/no). The outcomes are assigned to two values, 0 and 1. A Bernoulli process is obtained by repeating the same Bernoulli trial, where each trial is independent. If the outcome assigned to the value 1 has probability  $p$ , it can be shown that the summation of  $n$  Bernoulli trials is binomial distributed  $\sim$  Binomial( $p, n$ ).

as nuclear power generation become more focused on risk-informed application, the nature of the questions change – sometimes dramatically. And the NPP types change, thereby complicating analysis being performed with outdated tools. Then, the details of these conventional approaches are raised in prominence, frequently to the dismay of decision makers when evaluating the safety case.

The details that are produced from simulation approaches have been criticized due to the analysis computational burden and the resulting volume of information that can be produced. While not readily apparent, we should view these criticisms as potentials for enhancing accuracy and timeliness. For example, the scenario detail that is obtained as part of simulation analysis may be (if we ask in the right way) viewed as providing information not just on failures (the typical question) but on degradations, operability issues, maintenance issues, and human performance. Further, these simulation information streams may be mined for positive aspects of performance (what works and why) since the bulk of these simulated realities will not result in undesired outcomes – the “insights” are in the details.

While the static event-tree/fault-tree (ET/FT) approach has been used in the reliability modeling of systems for many years, numerous concerns have been raised about the capability of the ET/FT approach to handle dynamic and physical-based systems on a stand-alone basis. The ET/FT methodology does not treat the time-dependent interactions between physical processes and triggered or stochastic logical events as an accident evolves which may lead to coupling between these events through the control system. Even if these dynamic interactions are semi-quantitatively modeled through a classification of changes in process variables (e.g. "small", "moderate", "large"), it may lead to the omission of some failure mechanisms due to inconsistencies in the definition of the allowed ranges for the process variables. Such limitations and inconsistencies of static PRA with respect to system dynamics would be particularly important for complex systems.

We have stated the need for simulation. It may then be questioned as to what it will take to perform safety-related simulation – what tools do we need? Fortunately, a prototype simulation tool is available for use and customization at the INL. While we do not describe the details of this tool [called the Event Modeling Risk Analysis using bLock Diagrams (EMRALD)], Appendix A contains an overview of this event simulator tool.

## **2.3 Physics-based Simulation**

When simulating accident scenarios as part of a safety analysis, we will have the need of several physics-based simulations that must be run for one or more scenarios. A subset of these simulation modules might be run "offline" and their results stored in whatever format is native to that particular application. Alternatively, we may be able to translate these complicated mechanistic calculations into what are called “emulators” wherein the emulator mimics the more complicated analysis but is able to run orders of magnitude faster.

Let us describe a possible PRA scenario to better understand how physics-based simulation is used in the advanced PRA approach.

We construct a model representing the various structures at my target NPP. Then, as part of the simulation, we are going to represent a seismic event (which occur stochastically and with different magnitudes) and look at implications to the NPP structures. For a given earthquake that is “produced” by the EMRALD code, we query the results of the structural analysis (which could be a



load-capacity calculation using stresses and strains or an emulator-based calculation) in order to interpret the calculation into a state such as “no damage,” “cracked,” or “disabled.”

The simulation then continues by translating the physics-based mechanistic calculation into an impact in the accident scenario. For example, if the structure is cracked, this state would be applied to the component in the model (perhaps it is a wall or a pipe) using another stochastic model (in this case, a cracking model). Once the component state is specified, then the scenario would continue since the cracked component may experience a dislocation (the crack grows) or further damage. If the component were a pipe containing water, then we might experience a flow out of the pipe at the point of the crack, which could be in a critical location in the NPP. The water outflow from the pipe would immediately result in two impacts:

- Reduced flow to the pipe (possibly reducing heat transfer at the point where the water was originally needed)
- Possible spatially-related damage, depending on the location of the leaking water.

How we simulate these spatial types of interactions is through physics-based 3D environments that have been developed for industries such as visualization (e.g., movie special effects) and environment depictions (e.g., virtual reality). These 3D environments are capable of mimicking realistic physics such as flowing water and objects impacting other objects. For example, in the case of a pipe containing water, the environment model will know what the pipe material is, how fast it is going to impact items around it (if it cracks and breaks loose), its impact orientation, etc.

While the special interactions are being represented in the 3D environment, the accident scenario generator continues since water flowing from the leak may (later in time) fail collateral components (say a pump in the same room as the leaking pipe). At this point in the scenario, we are representing a flooding scenario (that was initiated by a seismic event). Further, there may be other components in that room that are sensitive to the leaking water, for example the pump motor controller which is an electronic component.

Note that this example accident scenario described above is just one possible outcome of the seismic event. However, it is the coupling of probabilistic and mechanistic calculations together that will, in the advanced PRA, be able to search (automatically) for vulnerabilities. Further note that a variety of special-type scenarios may be modeled and represented in these advanced models, including fire propagation, physical damage, flooding, and seismic impacts. A variety of 3D physics toolkits are available, both commercially and through open source.

One 3D toolkit that was evaluated for the development of this report was the “PhysX” environment. This analysis tool has several features, including:

- Discrete and continuous collision detection (to know when something hits something else)
- Solvers for rigid body dynamics (to mimic realistic movements)
- Fluid, particle, and character controllers (to represent fluids, movement of objects, and representation of people)
- Articulated mechanical dynamics (to represent complex components and systems)
- Fluids allow the simulation of liquids and gases using a particle system and emitters.
- Particle behavior permitting the simulation of explosions and debris effects

These types of environment modules will be able to be run as part of the cloud-based analysis server.

## 2.4 On the Need for Emulators

The SMR PRA framework discussion has identified attributes of the safety case that need to be considered in planning for development of SMR analysis capabilities. One overarching point is that a large volume of analysis is required; therefore, either an unprecedentedly fast and powerful analysis code will need to be developed, or a less fast but still powerful tool will be needed to underpin the development of one or more emulators for purposes discussed below. And, the development of the regulatory safety case is not the only driver of analysis – analyzing the performance case will also be important for SMRs.

### 2.4.1 Physical Modeling

When simulating accident scenarios as part of a safety analysis, we may have the need of several physics-based simulations that must be run for one or more scenarios. A subset of these simulation modules might be run “offline” and their results stored in whatever format is native to that particular application. Alternatively, we may be able to translate these complicated mechanistic calculations into what are called “emulators” wherein the emulator mimics the more complicated analysis but is able to run orders of magnitude faster. Note that a variety of emulators are available (from simple to complex).

By “emulator,” we mean a tool that mimics an analysis code by providing at least some of its key outputs, much more quickly than a code run, given a subset of the inputs to the full analysis code. Emulators have been used for generations, and better ones are still being developed. So-called “response surfaces” were being used in nuclear safety in the 70’s; since then, such things as neural nets and, more recently, Gaussian Process Models have been explored for application. The general idea is that one invests up front in doing enough code runs to characterize system response within an issue space, and “trains” the emulator with these results. Thereafter, one can get an estimate of system performance from the emulator without running the code itself, at a time savings that is enormous compared to the execution times of legacy codes. Emulators enable kinds of uncertainty quantification and vulnerability search that are essentially impractical with slow (traditional) codes.

### 2.4.2 Modeling and Analysis in the Licensing Process

Details of future licensing processes are evolving, but it is clear that SMR licensing will require:

1. A scenario-based approach to demonstration of plant capability;
2. Considered selection of events to be analyzed;
3. Graded approach to analysis of those events;
4. Significant attention to analysis uncertainty;
5. A PRA;
6. Resolution of SMR-specific technical issues;
7. Resolution of technology-specific issues;
8. Careful definition of system configurations and performance levels that need to be maintained during operation.

The first item above corresponds somewhat to the traditional safety analysis, but is formulated to allow for different technologies, and for certain lessons that have been learned in traditional licensing practice. In traditional licensing, selection of events to be analyzed has been partly prescribed, but

licensing logic generally depends on interpreting results in the context of the limiting case, and in general, significant effort has had to be expended in identifying limiting cases within the applicable issue spaces.<sup>4</sup> It is not generally clear what the limiting case is within a given issue space; many analyses need to be conducted in order to identify that limiting case. Results are not based on a single code run per design-basis event, but rather the whole suite of runs that were needed in order to establish which case was limiting. Moreover, determining which events need to be analyzed, and specifying the associated issue space (status of offsite power, number of postulated failures, postulated limiting initial conditions, etc.) also requires a significant amount of analysis in principle.

The second item marks a departure from the past conservative, margin-based approach to finding “adequate protection” in design-basis events. Lip service has been paid for years to “addressing uncertainty” for some purposes, but actual licensing analysis has changed only slowly and carefully in this regard. In past licensing practice, uncertainty was dealt with by requiring a set of analysis assumptions and analysis conservatisms that are believed to compensate sufficiently for analysis uncertainties to allow a finding of satisfaction of regulatory acceptance criteria, if analysis results so indicate. Put more bluntly: if the analysis says that the peak cladding temperature is below the limit, we can believe it, because the analysis includes diverse assumptions that drive the answer to higher values; therefore, the “real” peak cladding temperature is almost certainly less than the analysis result. In other words, “real” temperature < analysis result, and analysis result < acceptance criterion, so “real” temperature (must be) < acceptance criterion. One problem with past practice is that it is expensive; significant effort has been expended in the last decade and a half to allow plants to recapture some of this margin. Recently, many plants have done this using so-called “best estimate plus uncertainty” analysis. A particular interest of SMRs is that for fundamental reasons, they are expected to have greater effective margin than traditional designs. It is desirable to clarify this point, but difficult to do so within traditional analysis approaches.

The third item, PRA for SMRs, reflects relatively new regulatory requirements on new reactors,<sup>5</sup> and from a purely technical point of view, would include items 1 and 2. If NRC endorses consensus standards that apply to a given SMR design, the norms implied by that standard will need to be addressed. It is difficult to know whether consensus standards will be good enough for all vendors’ purposes.

The fourth item corresponds to SMR-specific issues, especially issues that pertain to most, if not all, SMR designs. Certain issues have already been identified for SMRs generally, including multi-unit-site considerations: how siting will work, what the required staffing levels will be at multi-unit sites, and so on.

---

<sup>4</sup> An “issue space” is a set of issues plus a set of conditions and assumptions within which those issues are to be analyzed. A familiar example is that of establishing a success criteria for a particular system following a particular initiating event. The issue space is defined by specification of the initiating event and other initial conditions (such as core history or whether loss of offsite power is lost concurrently with the initiating event). The general point is that analysis results must always be interpreted in context, and that context is here called the “issue space.”

<sup>5</sup> Part 52 requires PRA in design certification, and already-operating plants were required to perform IPEs; in addition, there is in 50.71 a requirement on new plants to perform PRAs:

(h)(1) No later than the scheduled date for initial loading of fuel, each holder of a combined license under subpart C of 10 CFR part 52 shall develop a level 1 and a level 2 probabilistic risk assessment (PRA). The PRA must cover those initiating events and modes for which NRC-endorsed consensus standards on PRA exist one year prior to the scheduled date for initial loading of fuel.

50.71 h [continued]

(2) Each holder of a combined license shall maintain and upgrade the PRA required by paragraph (h)(1) of this section. The upgraded PRA must cover initiating events and modes of operation contained in NRC-endorsed consensus standards on PRA in effect one year prior to each required upgrade. The PRA must be upgraded every four years until the permanent cessation of operations under § 52.110(a) of this chapter.

(3) Each holder of a combined license shall, no later than the date on which the licensee submits an application for a renewed license, upgrade the PRA required by paragraph (h)(1) of this section to cover all modes and all initiating events.

The fifth item corresponds to questions and issues relating to technologies that are qualitatively different from operating plants. This includes non-LWR SMR technologies, and could even include LWR SMRs sufficiently different from existing LWRs to call into question the applicability of existing rules. Passive system reliability falls into this category. SMRs will need to make exceptionally strong cases for passive system reliability, because SMRs achieve economy partly by not having certain active backup systems that the Westinghouse AP-1000 and ESBWR have. This would place significant demands on modeling and computation, even if assessment of passive system reliability were not still a research topic.

The sixth item relates to the implementation of the safety case during operation. This point should apply generically to all plant types, but is remarked here because the SMR-specific and technology-specific considerations driving the fifth and sixth points will play out in operation through this sixth item.

### **2.4.3 Evolution of Licensing Practice**

Reactor regulation has evolved significantly since most currently-operating plants were licensed. This has resulted partly from identification of weaknesses in traditional licensing and partly from improvements in modeling and analysis. As summarized below, further improvements are contemplated, even for the already-licensed technologies. For some beyond-Gen-II plant types, traditional licensing practice is even less satisfactory than for current plants: for example, certain analysis practices were adequate for analyzing certain events in large LWRs relying on active safety systems, but are unconvincing for other technologies (especially “passive” safety systems), and even for certain events in large LWRs.

Since future licensing practice has not been convincingly determined yet, there are limits to what can be said about it. However, based on experience in many technologies including nuclear, there are reasons to expect certain basic features in any reasonable future process. It will be necessary to show that hazards have been considered systematically, and appropriate controls have been developed. This will be done through a scenario-based analysis, with careful attention to uncertainty. In the parlance of traditional licensing practice, some (not all) of the “hazards” are design-basis events, and “appropriate controls” are single-active-failure-proof mitigation capability, shown through conservative analysis to satisfy acceptance criteria.

Although the design-basis approach has been criticized for decades, and (as noted above) regulatory practice has gone beyond it, there remains much to be said for including careful analysis of selected events in a safety case. In one sense, selection of events is inevitable, since it is impractical to analyze exhaustively every imaginable scenario in a complex technology. But downselecting to certain events is not just about analysis feasibility; if the case can be made that the capability to mitigate the selected events strongly implies the capability to mitigate all significant identified hazards, then a good safety case can be made. If it were really true that highly reliable shutdown and single-failure-proof mitigation of a few loss-of-coolant accidents addressed all scenarios, LWR safety would be a lot simpler. The trick is to choose the hazards (design-basis events, evaluation-basis events, ...) very carefully, and impose requirements on prevention and mitigation that properly balance cost and benefit.<sup>6</sup>

---

<sup>6</sup> Since WASH-1400, it has been suggested that the risk dominance of transients and small LOCAs relative to large LOCAs is related to the relative emphasis placed in design on large LOCA mitigation. Arguably, the situation is also related to the application of the same mitigation reliability criterion (the single-failure criterion) for all of these events, despite their very different prior probabilities. The single-failure criterion is arguably inadequate for transients. At least one of the so-called “TMI requirements” in 10 CFR 50 targeted this point, as did the IPE program. Refer to Attachment x to SECY 05-138 for a fuller discussion of this point.

NUREG-1860 presents a case for selecting the hazards based (at least in part) on risk analysis.<sup>7</sup> More recently (post-Fukushima), the NRC has issued a “Risk Management Task Force” report containing the following:

The inclusion of a design-enhancement category in the United States would result in the following framework for design-basis and beyond-design-basis events:

- Design-basis Events
  - Normal operation
  - Anticipated operational occurrences
  - Design-basis accidents
  - Design-basis external hazards
- Beyond-design-basis Events
  - Design-enhancement events
    - Internal events
    - External hazards
  - Residual risk scenarios
    - Internal events
    - External hazards

Design-basis events traditionally have been associated with mechanistic analyses, reliance on and protection of safety-related equipment, and the establishment of technical specifications and other licensing-related operational controls that have been deemed necessary for adequate protection of public health and safety. Beyond-design-basis events have more often included the use of best-estimate type analyses, PRAs, and in establishing additional plant protections to further reduce risks to the public health and safety and common defense and security (e.g., additional protections for station blackout conditions and aircraft impacts).

Two key concepts related to the identification of relevant scenarios, categorization, and subsequent design features and operating limits are:

- The threshold to define when a scenario needs to be considered within a category.
- The acceptance criteria to define when a design feature or operating limit provides the desired protection from the defined scenario(s).

The NRC’s Risk Management Task Force report considered the two-tier structure described by the U.S. Court of Appeals (i.e., adequate protection and additional protections), as well as the risk-informed and performance-based defense-in-depth concept used in the risk management goal and developed its Figure 4.2-1, which represents a general regulatory framework for nuclear power reactors (see next page).

---

<sup>7</sup> Quote from NUREG-1860: “In the current regulatory approach, risk information and insights are used to supplement the deterministic-based structure. In the [NUREG-1860] Framework, the regulatory structure is established from the start integrating both deterministic and probabilistic information and insights.”

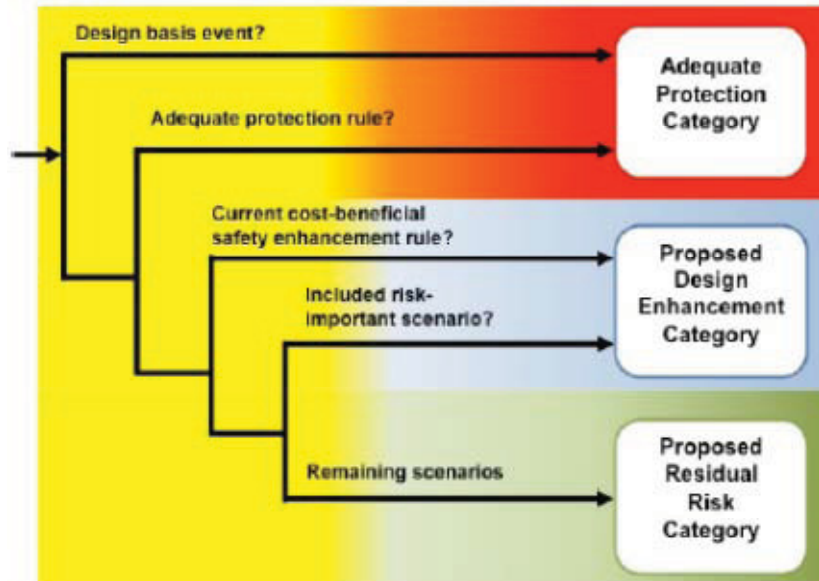


Figure 4-1 Regulatory Framework for Nuclear Power Reactors

For present purposes, it is neither necessary nor appropriate to assume that all details of the above excerpt will govern SMR licensing. **But it is clear that while terminology and details may vary, explicit, scenario-based analysis of plant safety will play a key role, and significant emphasis will be placed on selection of events to be analyzed.** Consequently, the need for extensive analysis, and emulators, supporting the safety case will be paramount.

For many reasons, including optimization of both production and safety aspects of the design, one needs a comprehensive model of system performance. For SMRs, it is desirable not only to demonstrate adequacy of the safety features, but also significant improvement in safety margin, in order to offset what are likely to be greater costs per installed kilowatt of generation capacity.

Even if the scope of considerations is artificially limited to safety and licensing, all aspects called out as “new” in the above excerpt will need to be analyzed in new ways with new analysis tools that have been calibrated to new experiments. Fortunately, development of new ways / new tools / new calibration approaches would not start from scratch; some development was occasioned as a result of the passive designs that have already been certified, and further developments have been undertaken to support the currently-operating fleet.

However, analyzing safety margin in SMRs from a modern point of view will require (1) a conceptual framework that is more nuanced than the existing regulatory framework, and (2) much better tools than those used on current plants.

This is not to say that these issues would justify a near-term investment in emulation. For now, the position is that the exhaustiveness needed in analysis to support the safety case will warrant both advanced simulation *and* emulation. Analysis to support the performance case may not initially require emulation, but would benefit from significantly improved analysis. As the state of practice advances in analysis and in emulation, we may reach a point at which emulation should be considered for performance as well.

#### 2.4.4 Analysis is a Key to Licensing

The technical case to be made for SMRs – licensing as well as economic- requires a great deal of analysis. The economic case may not immediately require a significant investment in emulation, but the novelty of SMR technology and the rigors of licensing will call for exhaustive analysis that could benefit significantly from a carefully-formulated emulation capability.

- Even for existing plants, a lot of analysis is required (more than usually realized). For example, “safety analysis” may seem to be predicated on one code run per design basis event, but a lot of analysis went into identification of that limiting case. Even now, errors or omissions are occasionally found in the identification of limiting cases for operating plants.
- Certain technology attributes of SMRs are actually trickier to analyze. This is particularly true of “passive system” reliability.
- Certain SMR issues are new and call for new analysis. For example, multi-unit issues will need to be thought about carefully, and (according to some), the matter of sharing operators between SMR units needs careful thought as well. This will need sophisticated analysis of the scenario set.
- Finally, licensing is more demanding than it was when the old paradigm was established; even without Fukushima, the bar would arguably be higher, and with Fukushima, doubt on this point is removed.
- Deployment of SMRs in multiple-use contexts (not just making electricity, or making electricity but in a situation that calls for load following) creates a class of analysis needs that current plants have not had to face.
- Moreover, given the inherent margin advantages that SMRs are believed to have, they arguably have an incentive to show that they can meet more stringent criteria.

In addition, the ability to call on analysis results in search algorithms will be hugely beneficial. This will help in:

- Deciding which events to analyze;
- Choosing the limiting case within a given issue space;
- Safety analysis;
- Analysis of plant performance.

In principle, these sorts of things can be accomplished by putting analysis codes into the loop. Historically, that has been utterly infeasible: the available tools were too slow and too unstable to be used in that way. In the near term, doing most of this will be difficult with foreseeable modeling tools. It can, however, be made practical with emulators.

## 2.5 Information Processing

We know that models and decisions will never be perfect. Nonetheless, it is important to define an approach to PRA that will improve the decision maker's ability to model complex nuclear systems probabilistically by showing:

- How best to use applicable models, data, information, and test results are available,
- How to understand their strengths and limitations in a probabilistic sense,
- How to determine the nature of the bounds they imply on key physical parameters.

Achieving improved decision making will require a novel approach to inference as it pertains to the collection and depiction of scientific and engineering information. The proposed SMR PRA effort will try to find practical, defensible tools and framework to be used in specific risk analysis situations.

Information (generally) reduces uncertainty. Even a significant "quantity" of information typically does not completely eliminate uncertainty, but only reduces it. From an experimental or operational viewpoint, when we collect and use data, we are transforming the data into information. We are learning which of several competing models is better, or increasing confidence in parameter values. This transformation takes place in conjunction with a model(s) and should include probabilistic information. These models require information for support, but the type of information varies from one model to the next. Real organizational value from data collection and analysis comes from the "processing, manipulating, and organizing" process when we obtain information.

In order to use probabilistic and mechanistic models, there are several hurdles to overcome as we construct a holistic advanced PRA framework:

1. Understanding of probability, especially the probability of the competing hypotheses. The translation of engineering and scientific information is the fundamental process for using probability.
2. The need to assess the probability of the evidence, conditional on each of the competing hypotheses. Note that this second item results in the need for multiple, competing models since no model is perfect.
3. When knowledge is available, overcoming inherent biases. It is critical to address a systematic method for mitigating biases while factoring in information, at the same time moving towards Bayesian inference and probability as the common language.
4. Representation of both causation and correlation as it pertains to physical processes.



## 2.6 Details of the Proposed Software Architecture

In this section, we describe some of the tools and approaches that may be used as the backbone of an SMR PRA approach.

Software Module	Description	Maturity Level	Open Source?
<b>SAPHIRE</b>	Software to solve static cut set based logic models	<b>High</b>	No, the source is available for use at the INL
<b>RELAP</b>	Software to solve T-H conditions	<b>High</b>	No, the source is available for use at the INL
<b>MySQL</b>	Software to manage data storage in a full relational database	<b>High</b>	Yes
<b>EMERALD</b>	Software to solve reliability-based simulation models	<b>Low</b>	Yes
<b>Jini</b>	Software to develop distributed systems consisting of network services and clients.	<b>High</b>	Yes
<b>WebGL</b>	Software to display advanced a graphical 3D environment in an Internet browser	<b>High</b>	Yes
<b>id Tech 4 Engine</b>	Software to create and use a graphical 3D environment and physics engine	<b>High</b>	Yes
<b>OpenBUGS</b>	Software to perform Monte Carlo-based Bayesian updating	<b>Medium</b>	Yes

The cloud-based architecture of the advanced PRA allows for multiple, heterogeneous servers running custom tools developed in multiple languages and on various platforms. Figure 11 shows that each integrated server typically runs a single analysis tool. As shown in the figure, these tools do not talk directly to one another (unless needed), rather they communicate with a central hub. This hub is responsible for receiving and storing input data (in a modern relational database) and analysis requests from the client. It also transforms data from its stored format into a format required by an analysis. The analysis provides an output that the hub will then receive and store or possibly transform and pass to another analysis.

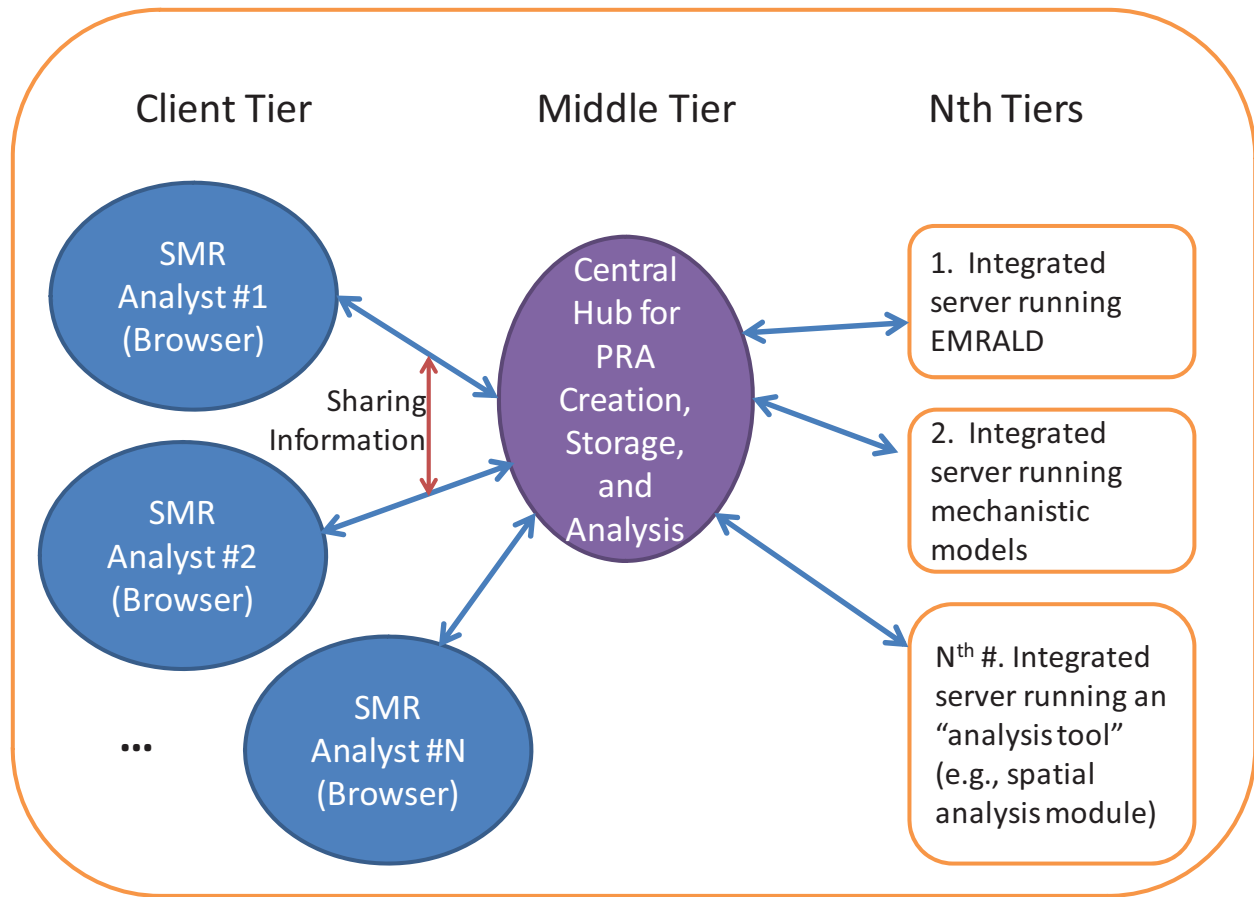
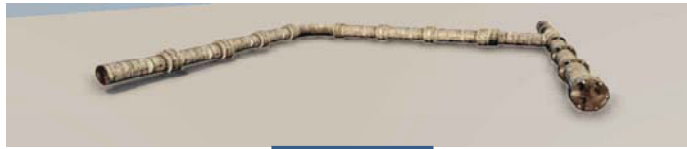


Figure 11. Schematic of the tiers of the cloud-based SMR PRA tool.

In order for these tools to communicate, each tool must define its required inputs and its possible outputs. For example, a seismic modeling tool may require an abstract equation representing the breaking point of a pipe and the magnitude of a given earthquake. The model then outputs at what point the pipe broke or perhaps that the pipe withstood the accident.

The output of the seismic model is passed to the central hub and stored as the result to be either returned to the user as the result or passed as input to another mechanistic or probabilistic module. Figure 12 provides an example of the transformations that might be performed for a segment of piping in a NPP in order to store the component in a relational database and display it in a 3D rendering environment (for example, using WebGL and the id Tech 4 engine). Then, in Figure 13, we show how that same component might also be associated with a mechanistic module to perform fluid flow calculations.

Transform 3D picture of pipe to XML representation.



Transform

Simplified XML representation:

```
<pipe>  
  <material>  
    <properties length = '22' metal='iron'>  
  </properties>  
</material>  
</pipe>
```

Figure 12. Example of an abstraction and storage of a physical component (piping).

Transform 3D picture of pipe to mathematical equation for flow



Transform

$$q - w = \left( u_2 + gz_2 + \frac{p_2}{\rho} + \frac{v_2^2}{2} \right) - \left( u_1 + gz_1 + \frac{p_1}{\rho} + \frac{v_1^2}{2} \right)$$

Figure 13. Example of the physical component (piping) coupled to an engineering flow model

The central hub of the cloud-based PRA approach must be developed with the ability to transform the results of one analysis into the required input format of another analysis. This transformation will require knowledge from the developers and/or subject matter experts of each of the individual tools to describe what the required inputs are, what format the inputs need to be in, what the output will be, and what format the output will be provided in for each module. However, once these are defined, they will be the same for all users of the tool.

Each analysis tool that participates in the system will have to provide this information about the input and output. Once the input and output of any system is known, a transformation can then be built to take the data as it exists and manipulate it into the form required by the receiving system. The output that the system provides can also be received and stored as it is provided or transformed to a format to be stored by the hub until it is needed to be provided to another system or as an output to the client (as illustrated in Figure 14).

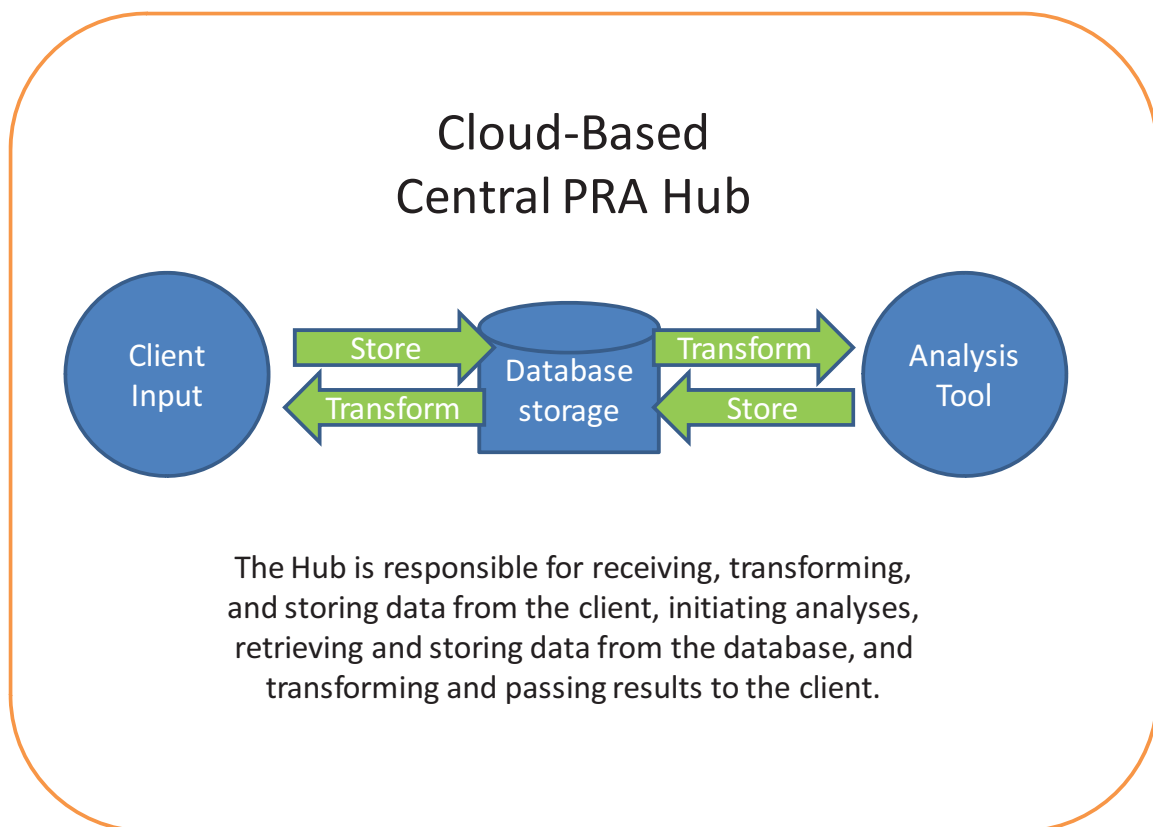


Figure 14. Operations of the "hub" portion of the cloud based SMR PRA tool.

The transformation can be developed in any software language but may be affected by the technology used for communication between the hub and the analysis servers. Gathering the requirements to build the transformations will be important.

The "n-tier architecture" we have described has a client, a middle-tier "hub", and various servers. One question is "how do these tiers communicate?" There are many different well-known approaches for communication including .NET, Java EE, RMI (Remote Method Invocation) which is included in Java EE, and Jini (also called Apache River) to name a few. In addition there are

commercial integration software packages that can help with the communication between legacy software applications.

.NET is a Microsoft framework and lends itself nicely to PCs but not as easily to servers running other operating systems such as Linux or Mac OS. However, web services are readily implemented in .NET as well as Linux for the purpose of integrating disparate heterogeneous software applications. A related tool is the Windows Communication Foundation. It is part of the .NET framework and supports applications built on other technologies that support standard Web services.

Java EE is built on top of the Java SE platform. It provides an API and runtime environment for developing and running large-scale, multi-tiered, network applications. Because it is built on top of Java, code developed can run on one platform and does not need to be recompiled to run on another. Java EE requires a Java EE server also known as an application server. These application servers are implemented primarily in Java and are also available for platforms from Windows to Linux and are available as commercial licensed products and open source products.

Jini technology is a service oriented architecture that extends and exploits Java to enable the construction of secure, distributed systems of network services and clients. Jini is comparable to the service-oriented architecture concept and provides a set of specifications as well as an implementation. The main parts to a Jini scenario are the client, the server, and the lookup service. Jini is available through an open source license.

One of the key items related to the advanced PRA will be in implementing the cloud-based framework. Fortunately, many application platforms are moving toward this approach and the quality and quantity of available software tools for development are rapidly increasing.

Hosting of the cloud-based PRA approach will be a series of servers. Example of the hardware architecture includes items such as Tesla-base cluster(s) (1792 threads per “computing processor”) where the peak performance is measured in Teraflops are available for a couple thousand dollars.

The traditional method of risk assessment has been the development of static models of a system (i.e. nuclear power plant) based on initiating events, event trees, fault trees and basic event probabilities. Over the years models have been created to help in understanding the risk of system failure. However, there are certain issues that static modeling do not adequately quantify. Some of these issues are how time of system component failures might affect risk. A dynamic simulation model of the system can take into account the timing of accident events.

Considering the completeness of current static PRA models including the probabilistic information that is contained, it is very advantageous to use these PRA models to auto-generate to the complexity of the system in a dynamic simulation model. This section describes a methodology to generate a dynamic model being developed from the static PRA models.

### **3.1 DYNAMIC SIMULATION MODEL**

The dynamic simulation model developed at the INL uses a modeling technology known as discrete event simulation. This simulation model maintains a dynamic list of events in time that are processed during the course of the modeling. The timeline events are created from analysis of stochastic variables that describe possible events in terms of a probabilistic distribution. The INL's risk simulation tool described in this report is called EMERALD.

The simulation model is based on several interconnected data objects. These data objects include:

1. **Simulation Object** – A simulation object is a subsystem, component, action or entity that makes up the modeled system. These modeled objects can be interconnected through parameter or attribute values. They become the basis of the modeling effort as each are evaluated for changes in state.
2. **Simulation States** – Each simulation object is further defined as to the possible states that the object can be in. These object states could be simple such as On, Off or Failed or might contain a complex series of states that might describe a decision path example.
3. **Simulation Events** – Events define a transition from one object state to another. This event transition is defined using various types of probabilistic distributions, object parameter trigger points or dependencies on other events. It is these events that are evaluated along the simulation timeline.
4. **Simulation Outcomes** – A simulation state might be associated with outcomes of interest in the system model. Certain outcomes might terminate the simulation. Recorded outcomes over several simulation runs become the basis of risk assessment and evaluation.
5. **Other Simulation Data Objects** – There are several other data objects that define things like required resources, variates and equations that support the simulation process and provide a way to conduct “what-if” types of studies.

## 3.2 STATIC PRA MODEL

The static PRA models in SAPHIRE consist of initiating events, event trees, fault trees, end states and basic events. Initiating events define a transient producing event that might have a negative impact on the modeled system. These initiating events are given frequency of occurrence which in a risk sense is hopefully very small. Each initiating event is the first node of an event tree that defines the top level events as a success or failure of safety systems or actions that are in place to protect the modeled system from damage. Each top level event in the tree is evaluated using fault trees that use Boolean logic of associated basic events to define the success or failure of a system. The basic events are defined in terms of probabilistic equations. Every logical path through the event trees is assigned an end state which in many models is binary in nature expressing a success or failure of the modeled system.

In all cases risk is a mathematical calculation expressing the probability that a specific success or failure path can occur. Critical failure paths can then be determined. Events that might be defined as a mean time to failure for example are reduced to a probability that it might fail during the time of the mission. Figure 15 shows an example fault tree, the simulation-based version of this fault tree is shown in Figure 16.

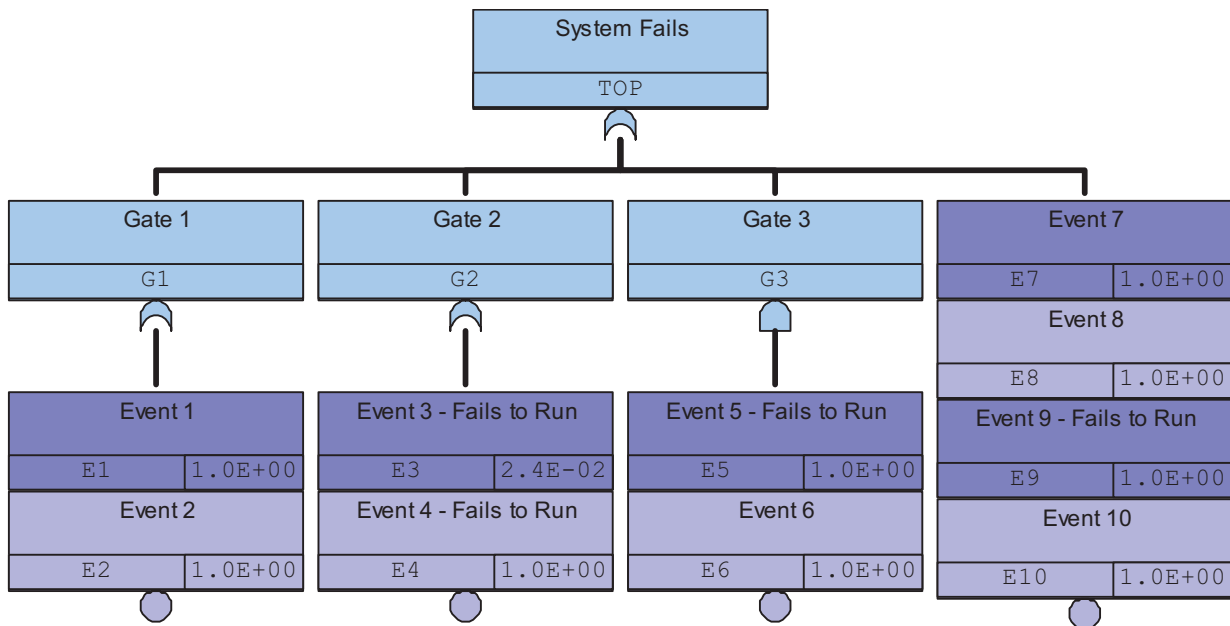


Figure 15. Example SAPHIRE Fault Tree.

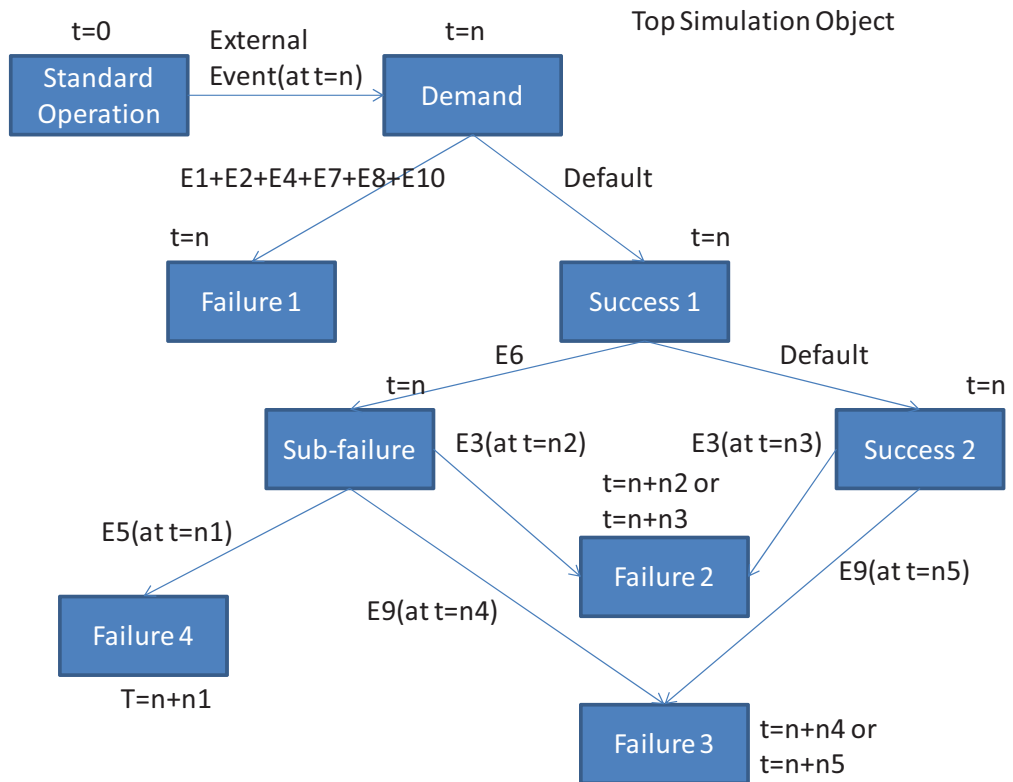


Figure 16. Simulation model representation of the simple fault tree.

The simulation model that is described in Figure 15 uses a “block diagram” approach to model building that is described in more detail in Appendix B.



**4.1 Interaction with System Analysis Tools**

Making the safety case for an advanced PRA requires a comprehensive, consistent and complete treatment of safety threats. This section describes the interaction of risk analysis tools (i.e., SAPHIRE and EMRALD) with mechanistic simulators of accident scenarios, hereafter referred to as “*System Analysis Tools*” (SATs). One of the most important drives for doing this is to extend from the deterministic margin into probabilistic loading versus probabilistic capacity, and in doing so, delineating and capturing uncertainty of all types and origins. Examples of SATs are either well-established legacy codes, like RELAP5 [1], TRACE [2], RETRAN [3], CATHARE, MELCORE [4], or any emerging new-generation system analysis codes (NGSAC [5,6,7,8,9]).

**4.1.1 Platform Technologies and Requirements for SATs**

Ultimately, the merging of PRA with SATs analysis tools must integrate the mechanistic methods of system process description with the probabilistic methods of reliability/risk assessment to provide a complete, consistent and comprehensive characterization of safety margins in a NPP. In other words, this enables “virtual plant safety testing”, where changes in plant conditions, operating procedures, applications of innovative elements (fuels, claddings, monitoring), etc., show their impact on plant safety margins.

The task of SATs is to provide detailed and credible answers to the stochastic discrete event simulator such as EMRALD about the outcome, or timing of events occurring in a postulated scenario. Thus, the SATs basic functions are to compute loads and to compute capacity on the specified SSCs at any given time of the investigated accident scenario progression.

One of the most important requirements for SATs is to enable adequate representation of stressors on a system. Stressors come in different categories: mechanical, thermal, irradiation and chemical, as illustrated in Figure 17. A particular stressor or parameter is considered important when it represents a challenge to integrity or operability of the SSC under consideration.

Figure 17 also shows modeling and simulation capabilities that the SATs need to possess in order to compute the loading. This includes simulation of physical processes (left cluster) and operating factors (right cluster), and modeling of the plant systems, which bring the physics and the operation into “loading”.

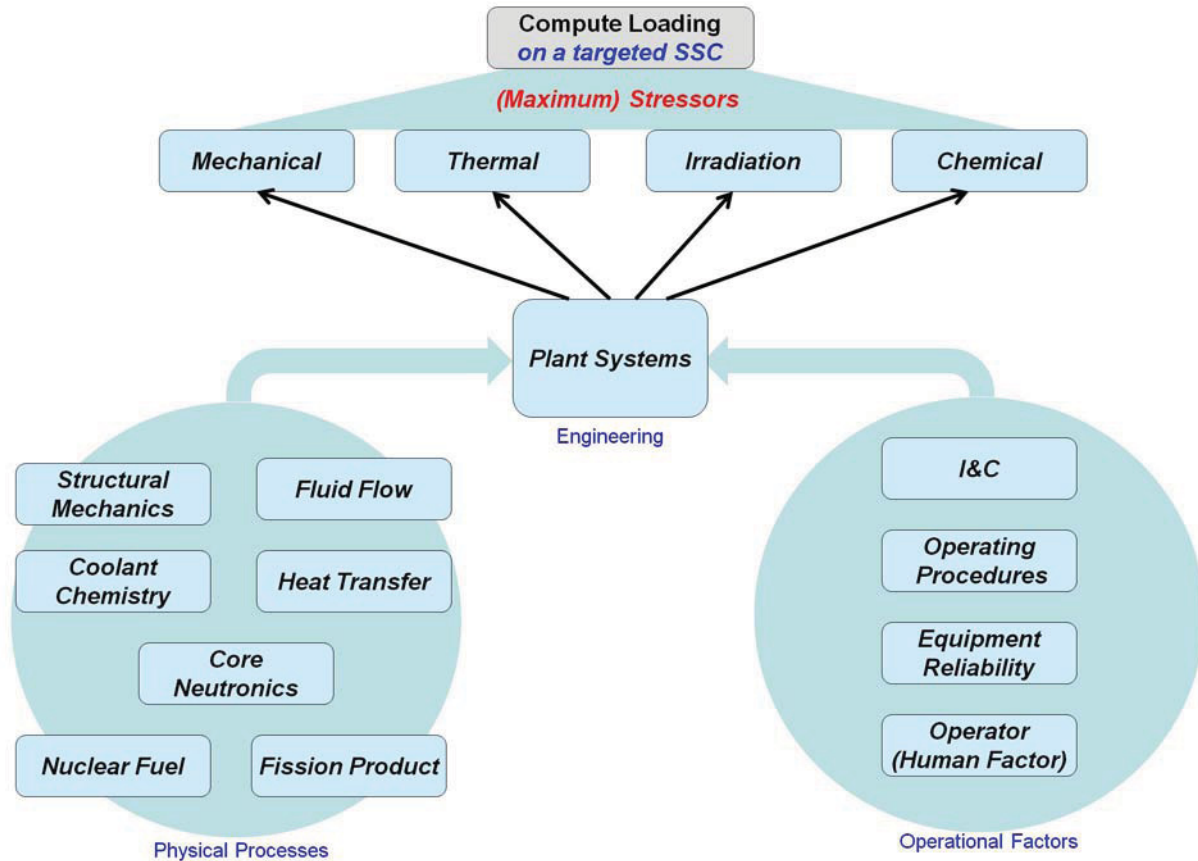


Figure 17: The SAT’s basic function “Compute Loading” and related capabilities. The driving physics are abnormal transients and accidents over short time scale with potential for peak loading.

When we couple probabilistic calculations with the mechanistic ones found in SATs, we then will be able to account for a variety of physics and a history of exposure to SSC stressors. Figure 18 shows the capabilities required to compute these types of SSC impacts. It can be seen that - in addition to simulation capabilities in support of loading computation - capacity computations require:

- “Damage calculation” capability, which translates stressors into material damages, including the synergistic effect of multiple stressors on damage initiation and growth; and
- “Degradation calculation” capability, which translates the material damage measures into degradation of structural materials strength, including the synergistic effect of multiple types of damages on degradation of material physical/functional properties.

It is noted that although the simulation capabilities under “physical process” and “operating factors” appear identical for loading computation and capacity computation functions, the actual emphasis and implementation of these capabilities differ for two functions. It is because the capacity computation function is concerned with long (days, or perhaps, years<sup>8</sup>) time scales of degradation processes, whereas the loading computation function deals with short (minutes to hours) time scales of accident processes.

<sup>8</sup> This is particularly relevant in the case of aging and plant sustainability safety analyses.

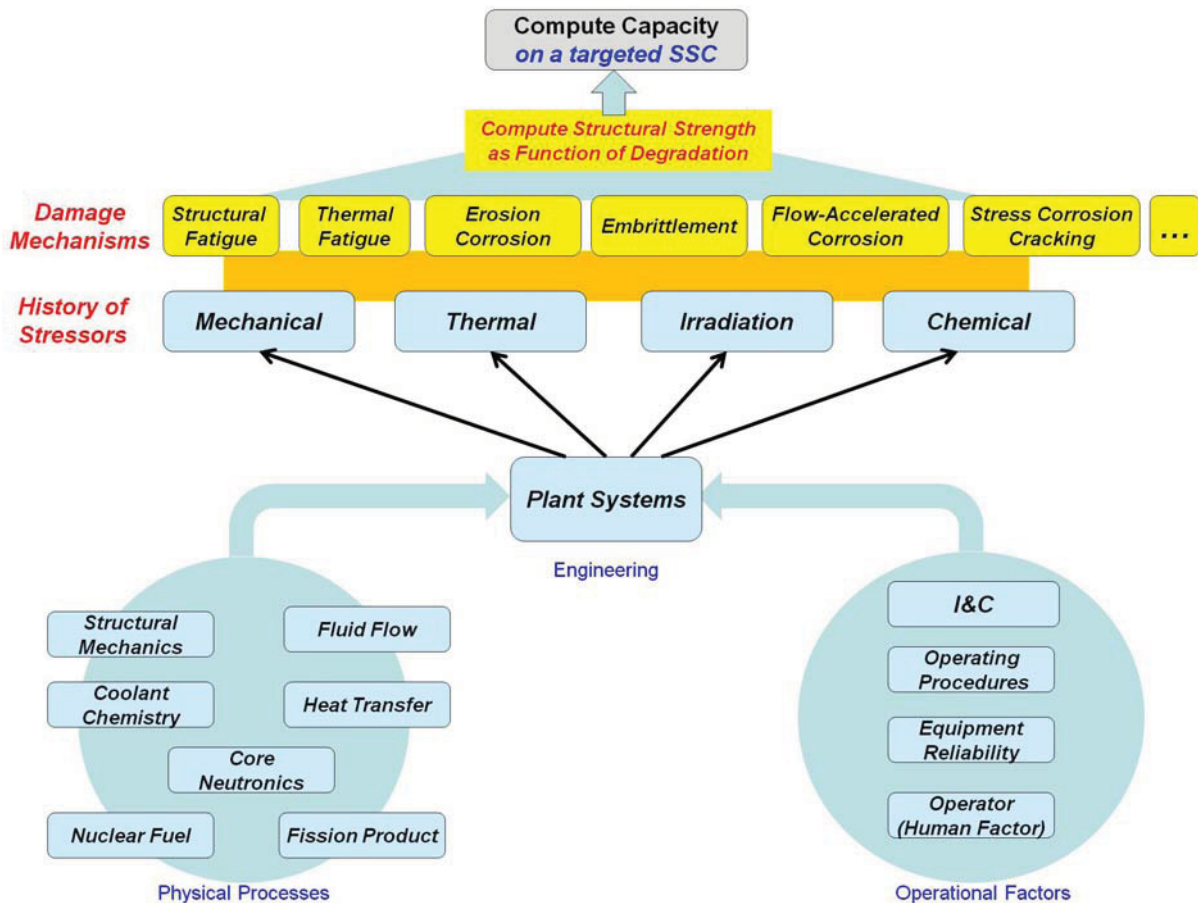


Figure 18: The SAT's basic function "Compute Capacity" and related capabilities. The driving physics is the degradation over long time scales.

There are a couple of circumstances when load and capacity overlap or are tightly coupled.

- When the loading computation requires the long "aging" result as an initial condition for transient analysis. In turn, the capacity computation for a real plant must also include transient loading analysis to capture operational, anticipated and abnormal transients that have occurred or been postulated to occur during the plant operation. Such abnormal transients are undesirable with respect to both plant safety and operability, as they often induce substantially more stresses on the plant's SSC, and sometime, these transients introduce stressors, which are not considered in the "normal aging" scenarios. Such periods of elevated degradation have a potential to accelerate the plant aging.
- When rapid degradation (capacity erosion) leads to a failure that alters the sequence. This is the case when a static, pre-calculated notion of capacity (structural strength) ceases to be valid (for example, in the case of spatial analysis described earlier). It is noted that in such cases the term "capacity" refers to intermediate margins, not the ultimate margin (capacity) targeted in the analysis. When "loading exceeds capacity" situation occurs for a SSC (which is not the defined search objective) during a transient, it leads to SSC failure. Modeling of failure, often fast-time-scale thermo-fluid-mechanical coupled processes, pertains to the domain of "physical processes" in Figure 17.

## 4.1.2 Main Concepts of the Mechanistic Implementation

A possible representation of the interaction between probabilistic and mechanistic process leading to an advanced PRA is outlined in Figure 19. The discrete event simulator (EMRALD) interacts with a PRA tool (such as SAPHIRE), to generate accident scenarios. While some of these scenarios can be analyzed directly by EMRALD, some would require detail mechanistic analysis of the non-linear engineering system response.

The SATs reside on a computational server such as a high-performance computing center. The interaction with EMRALD is implemented through the relational database SQL-type server. It places scheduled scenarios into the database, using a protocol specifically developed for interaction with SATs. These are essentially SQL tables, prescribing the scenario, and defining the expected results (in terms of, e.g. figures-of-merit). The information shall be pushed or pulled by querying these tables from both sides. Once new information is pushed – the signal shall be emitted to make the other side aware of updates. Open source tools such as the Jini technology described earlier may be used for the network control and message passing. Since a large volume of data may be generated from these simulation methods, methods to filter and classify data will be required – details of various methods to be considered are described in detail in Appendix C.

The main part of the computational server side is a **SAT controller**. The first thing the controller does is taking scheduled tasks from the SQL server, and making a decision on how to execute the task. This might involve the decisions on:

- Which code to use (some tasks might be more cost-effective using simplified but faster running calculations, e.g. RETRAN instead of RELAP-5),
- The level of fidelity (one might have multiple input decks, designed for different purposes, e.g. loss of coolant accidents versus feed-and-bleed or station blackout scenarios),
- Allocating computational resources (some heavy-duty tasks might require parallelization),
- Where and how to store the results,
- Data-mine looking for already-available runs (if a similar task has been done in the past, it might be more meaningful to just post-process it), and
- Whether the task is doable at all in a reasonable time frame using the available tools and computational resources. In the later case, the controller turns back “can’t do” flag for the selected task, in which case it is the job of EMRALD to make a decision on how to treat it.

Moreover, since there might be many uncertainties involved, either phenomenological, modeling, numerical or procedural – each scenario might require multiple runs, to enable some level of uncertainty quantification. It is desirable to avoid redundant runs if possible, but if this deemed necessary, the use of massive parallel computing is a possibility<sup>9</sup>.

The next step is to create the analysis tasks. Modern computational servers are typically designed to operate using *Message-Passing-Interface* (MPI) protocol [12], facilitating efficient multiple-task control. The other well-established tool for this purpose is available through the Qt framework’s [13] QProcess utilities, enabling high-level-of control in handling spawn runs. Once the task is launched, the controller needs to be able keep track of its completion (QProcess does provide these capabilities as well).

---

<sup>9</sup> One of the useful available resources for UQ is SNL DAKOTA toolkit [10], developed originally under the auspices of the NNSA ASC program [11]. The DAKOTA (Design Analysis Kit for Optimization and Terascale Applications) toolkit provides a flexible, extensible interface between analysis codes and iterative systems analysis methods (in the current context - SATs).

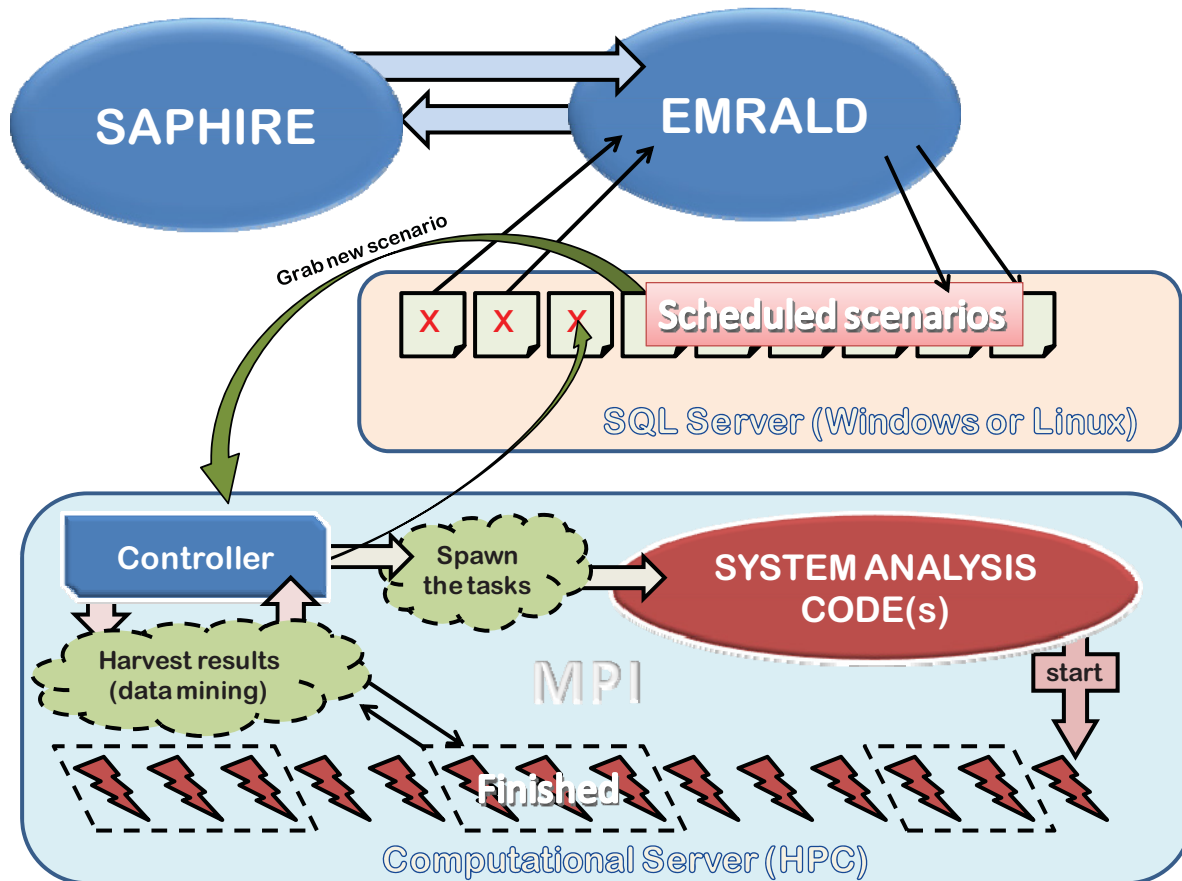


Figure 19: On interaction of discrete event simulator with SATs.

Upon a completion of the task, there are several steps to be executed. The first thing is to determine whether the run has finished successfully (i.e., no crash). This is particularly important for legacy codes, which tend to be less reliable due to outdated numerics<sup>10</sup>. (It can of course crash also due to hardware/network failures, especially acute for heavy-duty runs taking days to finish). In the case of a run failure, the controller must either decide to fine-tune the input deck (here, human intervention might be necessary, especially for legacy codes), and re-launch the task; or, in the case of multiple consecutive failures, to give-up and send a “cannot finish” signal to EMERALD.

If the run is successful, the controller would post-process the results, extracting the information needed for EMERALD. These data are sent back to the SQL server, marking the task as “finished”. Then, the results of the completed run are either archived in a suitable form for a future data-mining, or deleted to release computational resources. These should be decided upon the value and cost of the run.

<sup>10</sup> All known legacy codes (including RELAP5 and TRACE) are based on Implicit Continuous Eulerian (ICE) method, developed in late 1960s/beginning 1970s at Los Alamos National Laboratories [14, 15]. The ICE algorithm is based on operator-splitting between material and acoustic time scales, treating the later ones implicitly (typically, solving some sort of Poisson equation). In late 1970s, ICE was adapted for two-phase flow computations in system analysis codes, making its appearance in RELAP5, in the beginning of 1980s. From the point of numerics, all later system analysis codes (including the latest CATHARE and TRACE) were based on essentially the same algorithms, and inheriting the same limitations (mainly – stability problems). New generation system analysis codes [5] are based on fully-implicit algorithms [16], exhibiting significantly higher level of robustness.

Obviously, the controller by itself must be either multi-threaded or parallelized, enabling efficient continuous pre-/post-processing and data querying/mining. While, some of the tasks of the controller might need operator intervention, it is highly desirable to fully automate the whole process, which is possible using more stable and “smart” new generation system analysis tools.

It is also important to mention significant time scale separation between EMERALD computations and SATs. SATs are considerably more expensive to run; therefore, a great deal of selection must be executed on the EMERALD side, to keep the list of scheduled tasks limited.

### 4.1.3 Predictor-Corrector Feedback

The above-described strategy implies one-way interaction, i.e. there is no explicit feedback loop in the EMERALD — SATs interaction. The feedback is actually needed for at least some of the scenarios. For example, the SAT simulation might reveal the possibility for unknown scenarios, which could help EMERALD to refine its executions. While full coupling is complicated, it is feasible to organize “predictor-corrector” coupling. That is, after the first round of “SAT” answer (i.e., “predictor”), the EMERALD might correct the task/scenario, placing the “updated refined” task back into the queue, Figure 20.

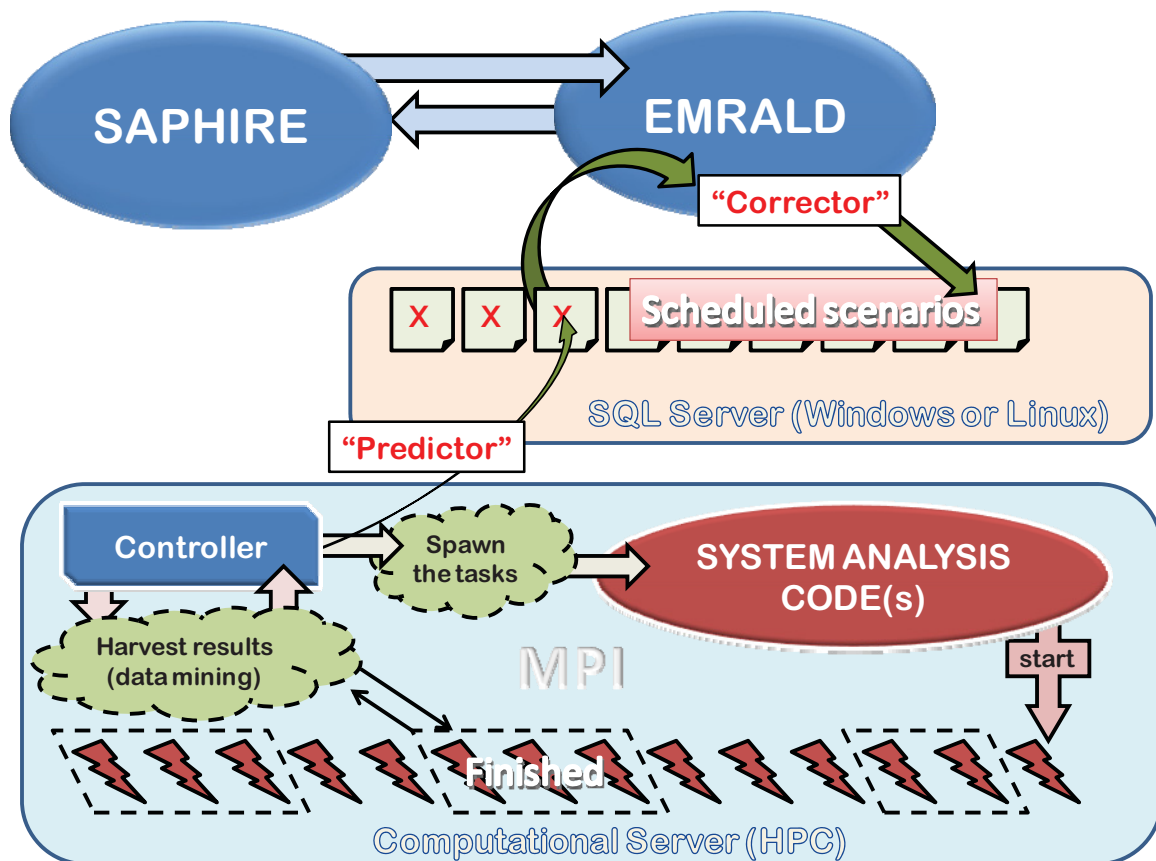


Figure 20: “Predictor-corrector” interaction of discrete event simulator with SATs.

## 4.2 Notes on Advanced Simulation Tools

**Code design and architecture.** To enable flexible system analysis tools, future advanced PRA tools using SATs should utilize modern computer science standards and capabilities, and in particular, the Object-Oriented Software design concepts [17]. It is very essential to be able to quickly modify SATs input decks, to reflect the application need, without major re-writing of the software<sup>11</sup>. This is especially acute when considering extensive and complex severe accident scenarios (like the recent Fukushima event), when continuous and quick user intervention might be essential to enable reflection of known changes in the progression of accident, and timely provide insights on the mitigation. One of these designs was recently developed and tested in [5]. Here, we will briefly highlight some of its major features.

The code design in [5] was based on the concepts of “**Components**” and “**Interfaces**”, Figure 21. **Components** are objects, which represent the pieces of system equipment. These could be pipes, pumps, pressurizers, reactor vessel, valves, steam generator, turbine, etc. These objects contain all the necessary geometry and equipment operational parameters, and also simulation models (possibly multiple, for different fidelity simulations). The models could be of different type, starting from simple ODE representation, 1D two-phase-flow thermal hydraulics, and going into the detailed 3D CFD-level representation (for T/H) or 3D nodal diffusion (for neutronics), Figure 21. The common feature of the component’s simulation models is their “solution” and “residual” vectors (obviously, both coming from the chosen underlying mathematical and physical models), and these are parts for building the “whole system” solution vectors and procedures.

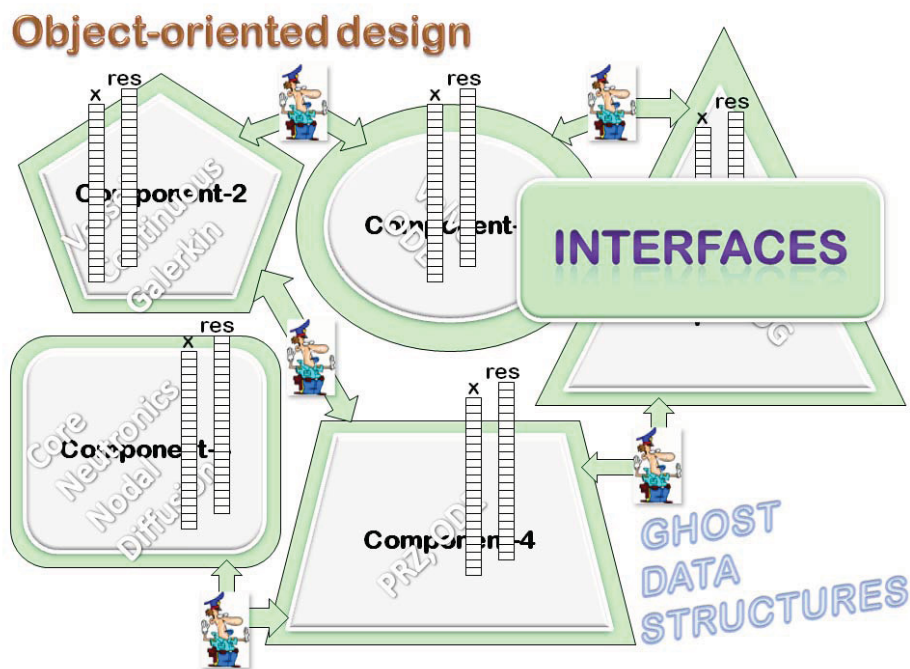


Figure 21: On using Object-Oriented Design concepts in building ASAT.

<sup>11</sup> Most of the legacy tools are designed with significant hard-wiring of the component models, closures and component connections.

**Interfaces** are also objects, which are specifically designed to enable connection between components. They typically do not solve anything (i.e., they do not have an explicit representation in the overall solution vector), but rather they are designed to serve as “traffic cops”: taking some information from one component, possibly process it (some sort of filtering, if it deems necessary), and pushing it into another component, Figure 21.

Using this plug-and-play “component/interface” concept, one can quickly assemble rather complex reactor system decks. Here, we show an example for OrSU APEX facility [18], in Figure 22. The shown deck contains over 150 components, connected together using over 200 interfaces.

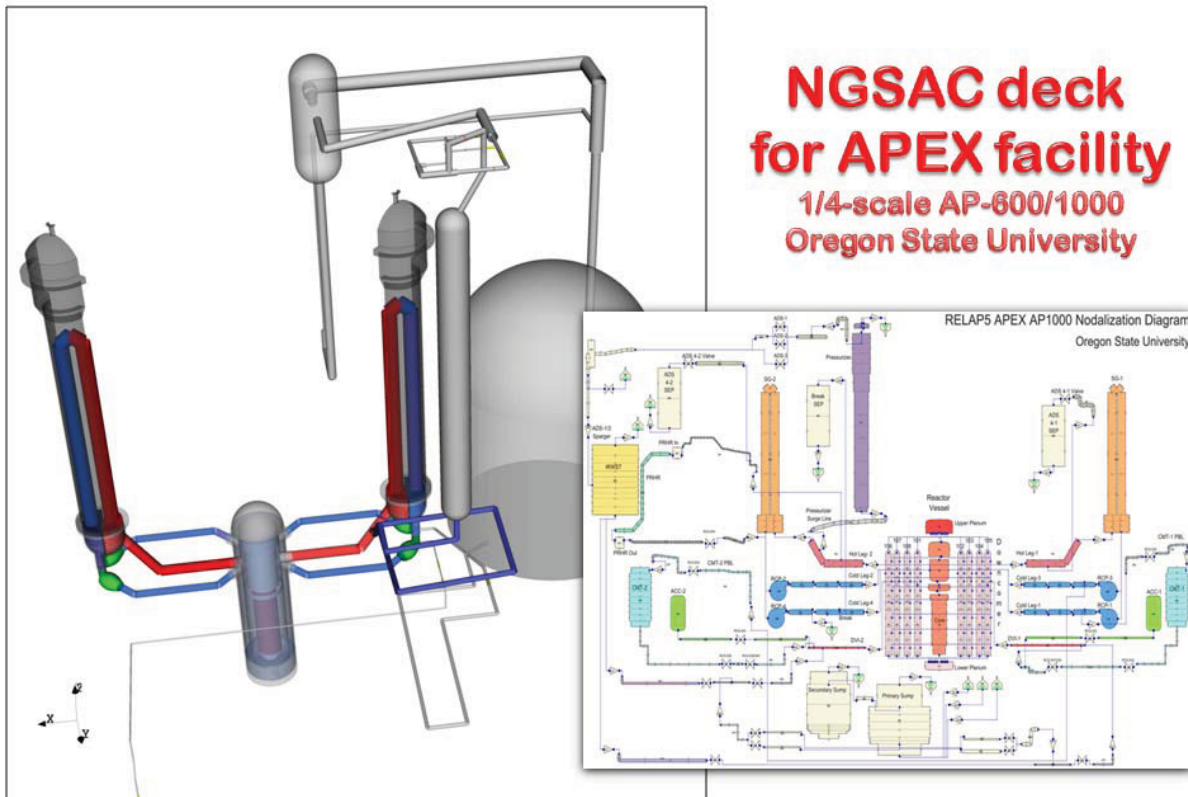


Figure 22: An example of using modern visualization concepts for building SATs input decks.



Designing with the “Component/Interface” code architecture concept enables to address the needs for “multiple fidelity” (both geometrical and modeling/algorithmic), and straightforward coupling with modern linear and non-linear solvers, Figure 23. As shown in Figure 23, one can easily extract the pieces of the solution ( $\mathbf{x}$ ) and residual ( $\mathbf{res}$ ) vectors from individual components, put them all together into global vectors, partition them<sup>12</sup>, and finally ship in an appropriate form either to ANL’s PETSc [21], or SNL’s Trilinos [2248] linear and non-linear solver packages. Typically, the simulation involves non-linear Newton method (“inexact” version of it), combined with GMRES Krylov linear solver [23]. One of the important features of GMRES is its ability to deal with non-symmetric matrices, in a Jacobian-free formulation [24]. Of course, the very important question is how to precondition GMRES. This can be achieved by using either math-based preconditioners (like ILU-based [23], those are part of PETSc’s KSP package), or different forms of physics-based preconditionings [24]. In the later case, the code design should retain a significant control over computational data structure, in order to implement algorithms like ICE from legacy codes, as a particular form of operator-splitting physics-based preconditioning.

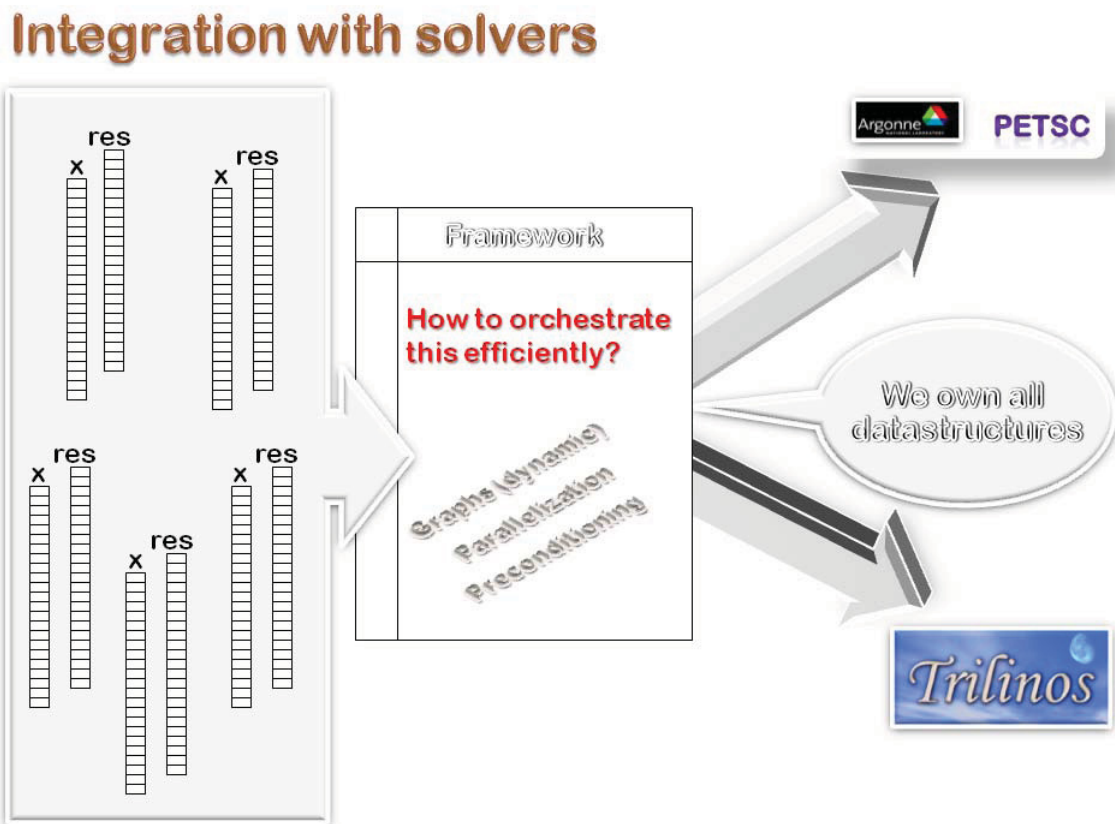


Figure 23: On integration with modern non-linear solver packages.

<sup>12</sup> For this purposes, one can use either ParMeTIS [19] (integrated into the ANL’s PETSC, and this is what was implemented in [7]), or Zoltan [20] (a part of SNL’s Trilinos).

**User interface.** Design of (graphical-) user interface is essential for modern system analysis codes. Providing a flexible and convenient way for an analyst to interact with software is extremely important not only from the point of efficiency, but also in order to reduce input deck errors, and for quick start-up and training. While a significant investment has been placed in the past in this area (e.g., NRC-supported Symbolic Nuclear Analysis Package, SNAP [25]), modern computational tools offer very convenient and efficient ways to rapidly develop flexible application-specific GUI/Visualization tools.

In Figure 24 and Figure 25, we show the major concept and a demonstration snapshot for a GUI of NGSAC [5]. The main underlying pieces of the GUI design are three open-source packages:

- Qt for rapid GUI programming [13,26],
- Visualization Toolkit (VtK) – for rendering 3D graphics [27], and
- Database server (SQLite) [28].

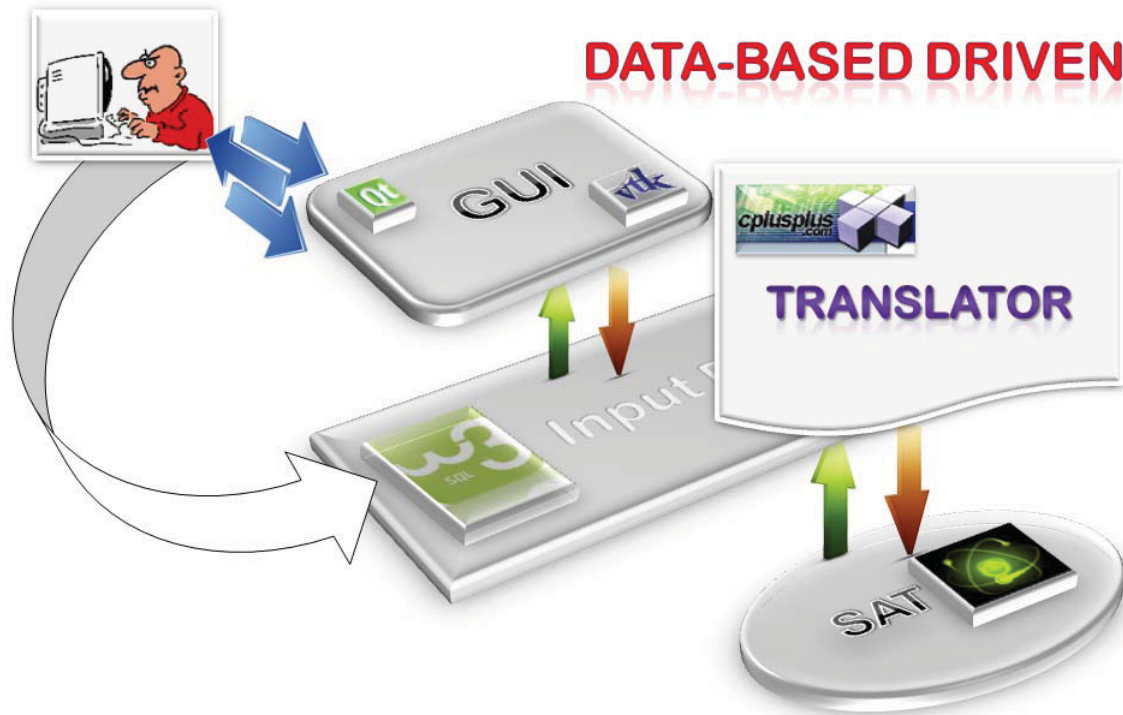


Figure 24: On designing user-interface with SATs.

As shown in Figure 24, we combined these three technologies, providing very convenient utilities for code developers to enable complete integration with the SATs. As one can see, in [5], INL has developed special C++ based utilities, integrated into each component's input method, enabling to define what shall appear in the GUI deck. Also, each component of the modeled system has special methods, which can be used to implement error checking and reporting.

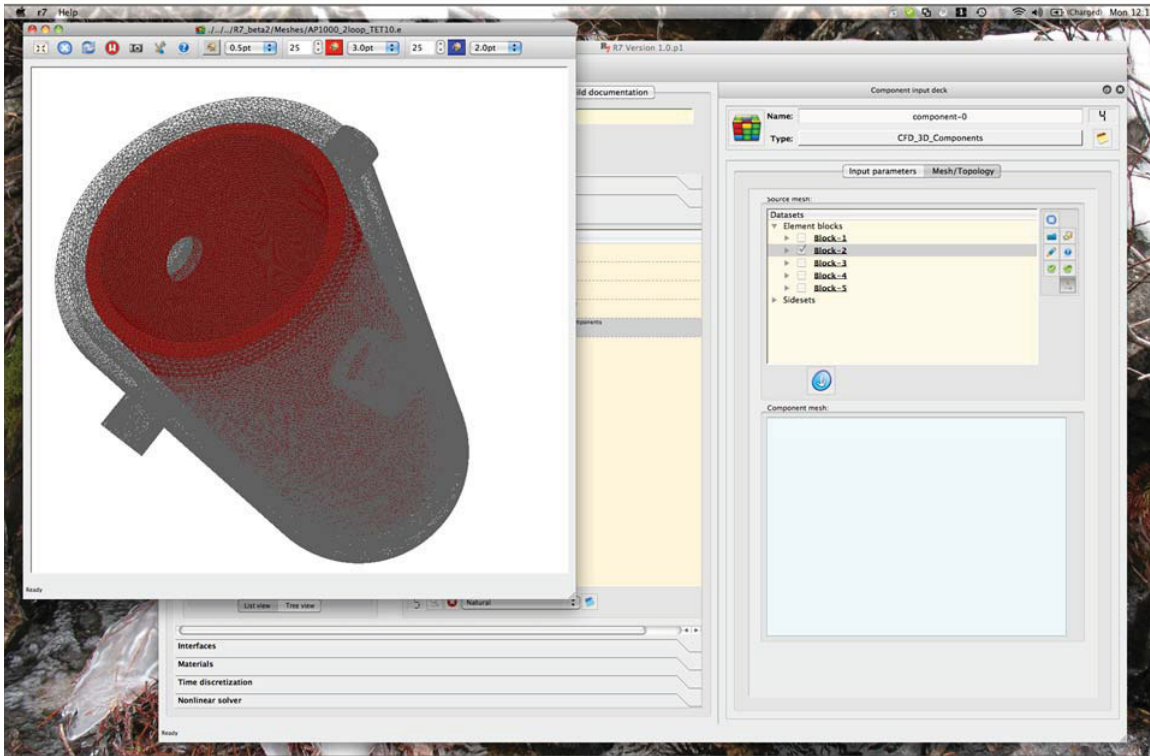


Figure 25: An example of using advanced GUI from [5].

Vtk provides a convenient and platform-independent way to render plots and 2D and 3D images, like those demonstrated in input deck, Figure 22, and 3D component mesh generation, Figure 25. Of course, one could also engage other convenient and more open source high-level visualization tools, like LLNL's VisIt [29] and Kitware's Paraview [30] (both based on Vtk).

## 4.3 References

1. RELAP5-3D Code Manual, Volume I: Code Structure, Solution Models and Solution Methods, INEEL-EXT-98-00834, Revision 1.3a, February 2001.
2. TRACE V5.0 Theory Manual. Field Equations, Solution Methods and Physical Models, Division of Risk Assessment and Special Projects, Office of Nuclear Regulatory Research, U.S. NRC, Washington, DC 20555-0001, [pbadupws.nrc.gov/docs/ML0710/ML071000097](http://pbadupws.nrc.gov/docs/ML0710/ML071000097).
3. J.H. McFadden, RETRAN-02: User's Manual, Energy Incorporated, Electric Power Research Institute, 1981.
4. R.O. Gauntt, MELCOR Computer Code Manuals, Sandia National Laboratories, U.S. Nuclear Regulatory Commission. Office of Nuclear Regulatory Research. Division of Systems Technology. 1998.
5. R.Nourgaliev, A. Bui, ..., N.Dinh,... R.Youngblood, etc. Summary Report on NGSAC (Next Generation Safety Analysis Code) Development and Testing, INL Report, Sept. 28, 2011, 359p.
6. G. Swindlehurst, Industry Requirements for the Next Generation System Analysis Code R7, EPRI Report, 1021085, December 2010.
7. R. Youngblood, R. Nourgaliev, D. Kelly, C. Smith, and T.-N. Dinh, "Framework for Applying a Next-Generation Safety Analysis Code to Plant Life Extension Decision-Making", ANS Annual Meeting, Hollywood, Florida (June 2011).
8. T.-N. Dinh, R. Nourgaliev, R.W. Youngblood, "A System Simulation Code to Support the Safety Case in the LWR Life Extension: Selection, Development and Testing of Code Architecture and Solution Algorithms", ANS Annual Meeting, Hollywood, Florida, (June 2011).
9. R. R. Nourgaliev, T.-N. Dinh, and R. Youngblood, "Development, Selection, Implementation and Testing of Architectural Features and Solution Techniques for Next Generation of System Simulation Codes to Support the Safety Case of the LWR Life Extension", INL/EXT-10-19984, Idaho National Laboratory Report (2010).
10. Adams, B.M., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Eldred, M.S., Gay, D.M., Haskell, K., Hough, P.D., and Swiler, L.P., "DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 5.0 User's Manual," Sandia Technical Report SAND2010-2183, December 2009. Updated December 2010 (Version 5.1).
11. Advanced Scientific Computing, NNSA, 1995, <http://nnsa.energy.gov/asc>.
12. W.Gropp, E. Lusk, A. Skjellum, Using MPI Portable Parallel Programming with the Message-Passing Interface, 2<sup>nd</sup> Edition, Scientific and Engineering Computation Series, 1999, Massachusetts Institute of Technology.
13. J. Blanchette, M. Summerfield, "C++ GUI Programming with Qt 4", Prentice Hall, Trolltech Press, 2<sup>nd</sup> Ed., 2008.
14. Harlow, F. H., Amsden, A. A., and Hirt, C. W., "Numerical Calculation of Fluid Flows at Arbitrary Mach Number," in "Proc. Int. Conf. Numer. Methods Fluid Dyn. 2nd," Berkeley, California, September 15-19, 1970 (LA-DC-11629).
15. Harlow, F. H. and Amsden, A. A., "A Numerical Fluid Dynamics Calculation Method for All Flow Speeds," J. Comput. Phys. 8, 197 (1971) (LA-DC-12190); AIAA Selected Reprint Series, Vol. 15, Computer Fluid Dynamics-Recent Advances (New York, 1973).
16. R.R.Nourgaliev, T.G.Theofanous, H.Park, V.Mousseau, and D.Knoll, "Direct Numerical Simulation of Interfacial Flows: Implicit Sharp-Interface Method (I-SIM)" (Invited Talk), AIAA 2008-1453, 46th AIAA Aerospace Sciences Meeting and Exhibit, January 7-10, 2008, Reno, NV, USA.
17. E.Gamma, R.Helm, R.Johnson, J.Vlissides, Design Patterns. Elements of Reusable Object-Oriented Software, Addison-Wesley Professional Computing Series, 1995.

18. K.B.Welter and S.M.Bajorek, APEX-AP1000 Confirmatory Testing to Support AP1000 Design Certification (Non-Proprietary), NUREG-1826, U.S. NRC, 2005.
19. Parallel static and dynamic multi-constraint graph partitioning. Kirk Schloegel, George Karypis, and Vipin Kumar. *Concurrency and Computation: Practice and Experience*. Volume 14, Issue 3, pages 219 - 240, 2002.
20. K. Devine, E. Boman, R. Heaphy, B. Hendrickson, and Courtenay Vaughan, Zoltan: Data Management Services for Parallel Dynamic Applications, *Computing in Science and Engineering*, 2002, 4, (2), pp.90—97.
21. Satish Balay, Jed Brown, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang, PETSc Web page, <http://www.mcs.anl.gov/petsc>, 2011.
22. The Trilinos Project, Sandia National Laboratories, <http://trilinos.sandia.gov/index.html>.
23. Saad, Y., *Iterative Methods for Sparse Linear Systems*, 2<sup>nd</sup> ed., SIAM, Philadelphia, 2003.
24. Knoll, D.A., and Keyes, D., “Jacobian-free Newton-Krylov Methods: A Survey of Approaches and Applications,” *Journal of Computational Physics*, Vol. 193, 2003, pp. 357, 397.
25. Symbolic Nuclear Analysis Package (SNAP), K.Jones, J.Rothe, and W.Dunsford, U.S. NRC, NUREG/CR-6974, 2009.
26. M.Summerfield, *Rapid GUI programming with Python and Qt*, Prentice Hall, 2007.
27. W.Schroeder, K.Martin, and B.Lorensen, “Visualization Toolkit. An Object-Oriented Approach to 3D Graphics”, 4<sup>th</sup> Edition, Kitware, 2006.
28. R.F. van der Lans, *The SQL Guide to SQLite. A Comprehensive Tutorial and Reference for the SQLite Open Source Database Server*, Lulu Publisher, 2009.
29. Visit Visualization Tool, Lawrence Livermore National Laboratories, UCRL-WEB-229972, <https://wci.llnl.gov/codes/visit/home.html>.
30. Paraview: Open-Source Visualization, Kitware, <http://www.paraview.org/>.

# Implementation Plan

To implement the SMR PRA framework, the INL will focus on the development of computational approaches and tools that will be used to conduct analysis of safety performance of NPPs. As a part of this approach, the use of PRA will be explored, wherein safety-related scenarios will be described via probabilistic models (e.g., fault trees, event trees, influence diagrams, simulation) with the intent to support risk-informed decision making.

Development and implementation of SMR PRA methods will require new analytic methods and adaptation of traditional methods to the new design and operational features of a SMRs. As an example, PRA has been used to evaluate the safety of nuclear plants with respect to severe accident consequences, such as core damage. However, these current methods have seen little application related to modeling the margin of safety. As such, there is a need to move beyond the current limitations of static, logic-based models in order to provide more integrated, scenario-based models based upon predictive modeling which are tied to causal factors. Understanding the causes that reduce safety margins will be the key to:

1. Developing engineering controls to effectively manage risks and
2. Demonstrating the technical basis of safety margins as part of the licensing phase.

An initial thrust of this program element will be for the construction of an overall analysis and modeling cloud-based framework that will be open for National Laboratory, DOE, NRC, and industry use in order to successfully demonstrate the technical basis related to safety margins. We will build this capability by focusing on available open source tools coupled with existing PRA capabilities available at the INL.

The overall proposed approach is broken down into three phases.

- Phase 1           Addresses the development of the general technological framework for the SMR PRA concept. Completion of this phase will result in the generation of proposed features and specifications for the supporting framework and analysis modules. This report is the first step forward in completing this Phase.
- Phase 2           Addresses the research and development required prior to trial implementation of the key modules that fit into the SRM PRA framework. Completion of this phase will result in (1) the technology evaluation and preliminary development of needed modules and (2) the determination of the overall open framework supporting the operation of the concept.
- Phase 3           Addresses the trial implementation of the SMR PRA concept demonstrating its usefulness as a design, decision, and optimization tool. Completion of this phase will result in the implementation, testing, and trial application of the concept.

Phase 1 of the project provides the genesis of the detailed SMR PRA concept. Since this project is an evolution of current PRA practices, the formulation of the technological framework is vital to success of the overall project. Included in the formulation of the framework is consideration of supporting technologies that are currently available (e.g., 3-D environments, multi-processor

computers, open source software) and technologies that may mature within a short time period (e.g., voice-controlled software interfaces) that could be of use to the U.S. nuclear community.

Phase 2 of the project provides the beginnings of the implementation for the SMR PRA concept. Of particular concern during this phase of the project is in the behavior of the key modules with respect to global interactions with the advanced PRA environment. One of the key drivers of the SMR PRA framework will be in the modularization of important parts of the supporting and analysis environment using the cloud-based approach. The work for Phase 2 is subdivided into two tasks.

- Task 2.1, the selection of supporting SMR PRA modules, focuses on defining the framework and relevant technologies needed for integration into advanced PRA.
- Task 2.2, the pre-development of the SMR PRA modules, focuses on determining specific attributes operational characteristics for the key SMR PRA modules.

Phase 3 addresses the implementation and testing of the key supporting and analysis modules for the SMR PRA framework. This phase results in a trial application and refinement of advanced PRA for a NPP design, optimization, or analysis problem.

# Simulation Using EMRALD

## A-1 Discrete Event Simulator

In order to develop a dynamic PRA capability, a prototype discrete event simulator was created as an open source tool that can model dynamic systems and generate probabilistic risk/reliability results. The development consisted of three major parts:

1. Development of a discrete event modeling software library
2. Design of a simulation object model for dynamic modeling
3. Construction of the graphical user interface (GUI) that will allow risk assessment analysts to more easily develop these models

### Simulation Object Model

A key part of the EMRALD tool has been to develop an object-oriented software model that is flexible enough to support the varied dynamic simulation models (e.g., fails to operate, fails on demand). An object-oriented approach better matches the simulation model to real world objects and makes it more intuitive for the modeler. The following table lists and describes these objects.

Simulation objects.

Object	Parameters	Description
Simulation Control	Name Version Description Time Iterations Real Precision Process Flags	A simulation model has one and only one control object. This object defines the model name, version and provides a place for the modeler to describe the model. It also contains the simulation controls which include the mission time or the maximum time of the simulation, the number of iterations that control the number of times the model is executed and the real precision value that controls the comparison of two real values for equality. The iteration control is very important in determining the accuracy of the simulation results. The lower the expected probability of failure the more iterations that will need to be executed for accurate results. As a rule the more iterations the better the result but that must be tempered with the computer's time to execute large numbers of iterations.



Object	Parameters	Description
Simulation Objects	Name States Initial State Attributes	Simulation objects can be physical objects of the simulation as well as abstract entities. They provide a way to capture information about the objects used during the simulation. Objects are identified by name. Operation states are defined for each object and the initial state of the object at the start of the simulation is indicated. The simulation objects transition between these states throughout the simulation. The path the simulation takes through the different object transitions is controlled by the randomness of the Monte Carlo methodology. Each object can also be given one or more attributes. An attribute carries a value associated with the object's current state and can be accessed by other objects of the simulation as indicated by the connections established between objects. States and attributes are further described below.
Object Attributes	Name Minimum Maximum Child Objects	Attributes hold values of the object. These values change based on the current object state and/or the sequence of state transitions. Each attribute is identified by a unique name for the object. A minimum and maximum value can be defined. Attributes are connecting objects in the simulation model and are defined as having a parent object and one or more child objects.
Resources	Name Count	A resource is a limiting factor for event execution. A simulation event might require access to certain resources in order to complete its task. Each event during execution will request those resources and if one is not available then event execution is paused and other events can occur. Once the resource has been released and is available for reuse, the task is able to continue its processing. The count is the actual number of these limited resources.
Variates	Name Distribution Inputs	A variate is a random variable or set of values determined by a probability distribution. A variate is used during the simulation to control conditional event transitions, the event's execution starting time and its duration during the simulations and the attribute values. The variate gives the simulation the Monte Carlo randomness needed to generate all of the probable paths of model execution. Each different distribution requires different input parameters to define the actual distribution shape. The available distributions include: Beta, Constant, Discrete, Erlang, Exponential, Gamma, Log Normal, Normal, Triangular, Uniform and Weibull.

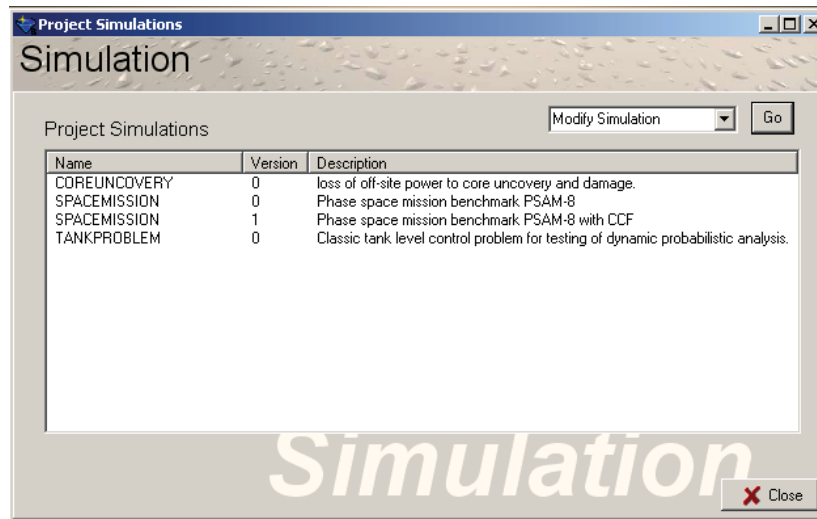
Object	Parameters	Description
Simulation Functions	Name ValueAt(Time) TimeAt(Value)	<p>Simulation functions provide a means of calculating attribute values during simulation using other attributes and simulation states including time and previous attribute values. The simulation function could support the calculation of object attributes that might include temperature, pressure, level or power to name just a few. The value of these attributes is modeled using a set of continuous functions. The various functions are dependent on the operation state of the various objects and simulation time. Each named function actually contains two functions. The ValueAt function returns the function value at the current or any future time of the simulation. The TimeAt function returns the future time, if valid, when the attribute will reach the value given. The TimeAt function is used to schedule events during the simulation based on simulation attribute values.</p>
Dynamic Outputs	Name Initial Value Time Step Data Type	<p>Dynamic output objects are used to store the dynamic results of the simulation. They statistically combine the results of all iterations of the model. The initial value of the output curve is specified along with the time step of each data point after that. Because of the limited storage space for the data the total time bins (mission time / time step) cannot be greater than 2000. The data type controls how the data is collected. The probability type records a single failure time or success for each iteration of the model. The series types use values from an object attribute and accumulate an average over all the iterations. The different series types control the way the data between the simulation events is interpreted, linearly or as a step function.</p>
Object States	Name Attribute Values Outputs Transition Events	<p>Every simulation object will have one or more states. A state is identified by a unique object name. The state plays a large part in the control of the simulation as the simulation transitions through the various object states. The state determines the objects' attribute values, records failure or attribute information in the dynamic outputs, and activates the next transition by entering events into the simulation event queue.</p>

Object	Parameters	Description
Transition Events	Name Type Delay Duration Required Resources Owner/ Current States Final States	<p>The transition event controls the simulation transitional path for each iteration. Events define happenings or processes that occur in time. Since events can transition several object states, the event name must be unique for the simulation. There are several types of events that are defined by how and when the event is created, when the event begins execution and whether the event terminates the simulation. Events are created and activated in one of four ways. They can be created at the start of the simulation during initialization; they can be created unconditionally by a transition into an object state; they can be created conditionally based on object attribute values; finally, they can be created randomly based on variate values.</p> <p>An event can be scheduled to start immediately or its start can be delayed randomly using a variate object. An event can also generate several copies of itself randomly through the duration of the simulation. An event can also occupy a constant duration in the simulation time space. Duration can also be specified as immediate or as a variate. Required resources can also be identified. Events usually change the condition of the simulation and this is reflected by the state transition arrow from the current object state to a new object state.</p> <p>Events follow the following processing sequence: Attributes are updated for possible time and attribute changes. If resources are needed they are requested and the event could pause until the resources are available. Next the event works its duration in time. Other events are allowed to run concurrently with this event while it works. Next the requested resources are released. Object states are changed and object attributes are reevaluated for the new states. Follow-on events are scheduled in the event's queue and output values are recorded.</p>

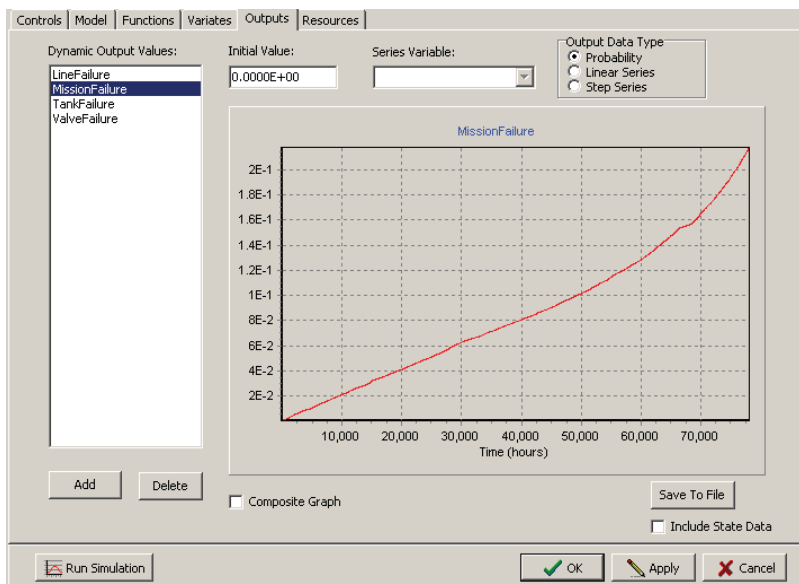
## A-2 Graphical User Interface

Key to meeting the goal of having risk assessment analysts able to do the modeling in the discrete event simulator is to design a GUI that is intuitive and easy to use.

The GUI is designed around the simulation objects described in the previous section. Designed models are stored in the data base and are easily created, accessed and modified. The first screen of the simulator (below) lists the available models. It lists the name and description from the control object.



From the simulation selection screen a user can select a current model for modification or execution, select a model for deletion or enter a new model. Adding or modifying a model will bring up the model editor and simulation screen (below).



The form is organized using several tabbed screens for the creation and modification of the simulation objects described earlier. All the tabs except “Model” are similar in that they allow for the simulation object addition or deletion in a list on the left side of the screen. Parameters of the simulation object are defined and modified on the right side of the screen. The bottom portion of the screen contains buttons to run a simulation, to save changes and to close the screen.

### Controls Tab

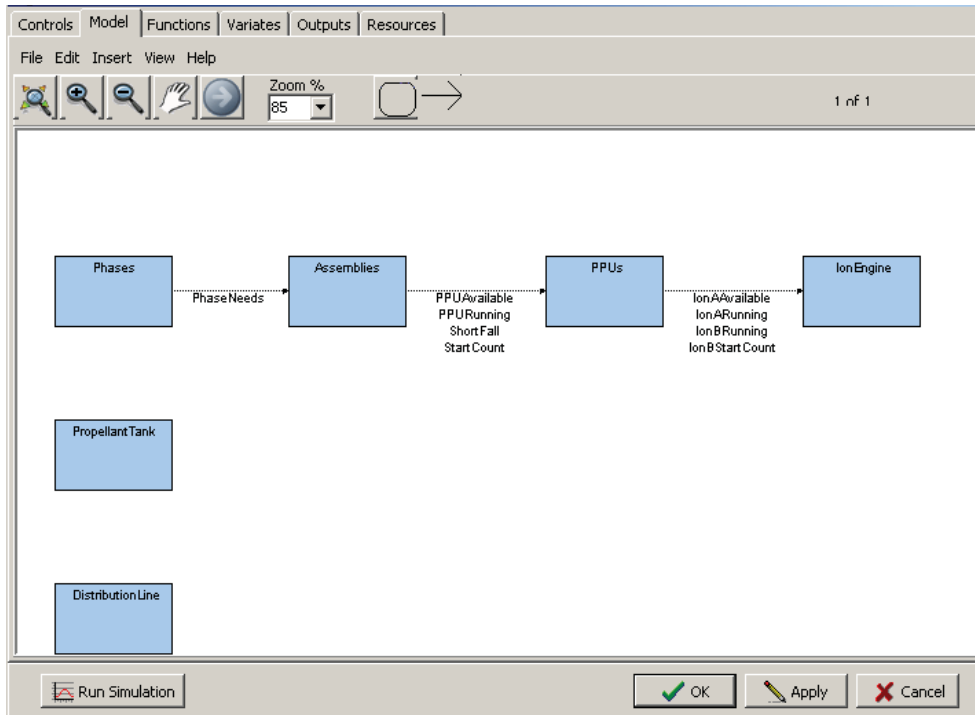
The “Controls” tab provides an area for entering the high level simulation controls (below). This includes the simulation name and description. Version information is controlled by the “New Version” button. A new version starts as an exact copy of the current simulation and increments the version number. Other parameters including the start and stop times, time step, time unit, total iterations and real precision value can be entered. The check boxes are not stored in the data base but can be used to record iteration history information to file and fix the random seed sequence for repeatable results.

The screenshot shows a software window with a tabbed interface. The 'Controls' tab is active, showing the following fields and controls:

- Simulation Control Variables:
- Name:  Version:
- Description:
- Start Time:  Iterations:
- Stop Time:  Real Precision:
- Time Step:
- Time Unit:  (dropdown menu)
- Use Fixed Random Seed:
- Record Histories:
- Record Failure Histories Only:
- Accumulate Statistics:
- Buttons at the bottom:

### Model Tab

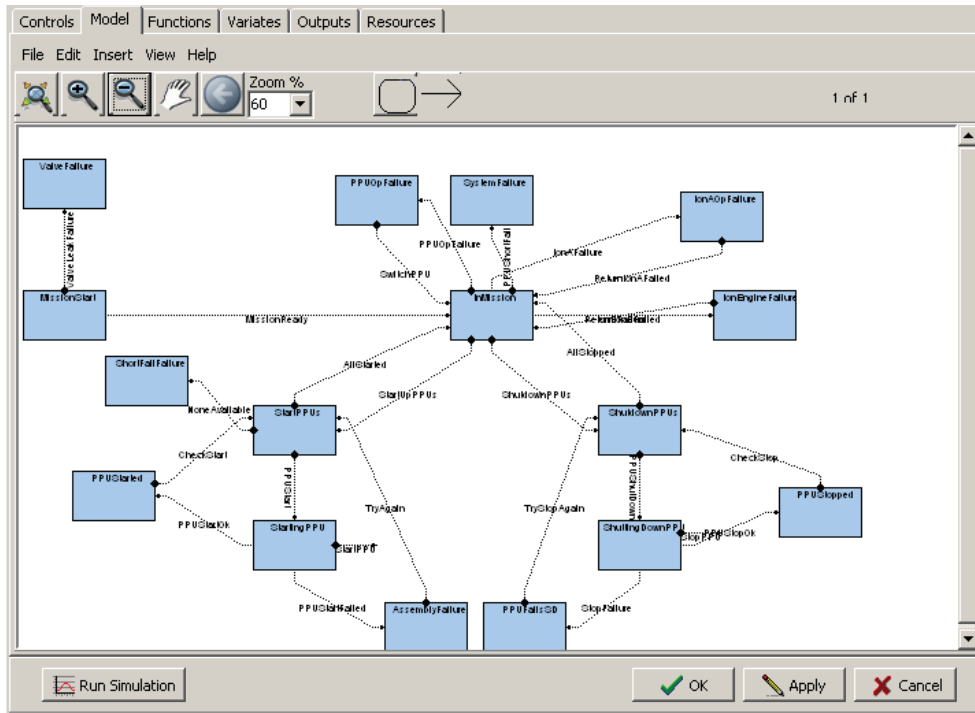
The “Model” tab provides a graphical canvas for drawing and designing the simulation model. It includes two primary views: objects and states. Every model includes one and only one object view (see next page). The object view displays the simulation objects and the relationship of the objects. The relationship is indicated by arrows and attributes associated with the arrows. The relationships between objects cannot be cyclic. Object attributes are visible to all of its children for use in functions and conditional events.



The model includes a state view for each simulation object in the object view (see next page). The state view displays the object's state diagram which includes the states and the state transitions that connect the states. The state transitions define the owner/current state of the transition and point to the next state of the transition. Each state transition must have one and only one owner state. The owner state is the one that will create and activate the transition event when that object enters that state. A transition event can also be assigned to a state to effect a transition based on entering another object's state which owns the same event by name.

From the state view the user can add new states and state transitions. The editing and modification of the model also is done from this view. The object states and state transitions for the most part define the simulation process by defining how changes in state affect the next state transitions.

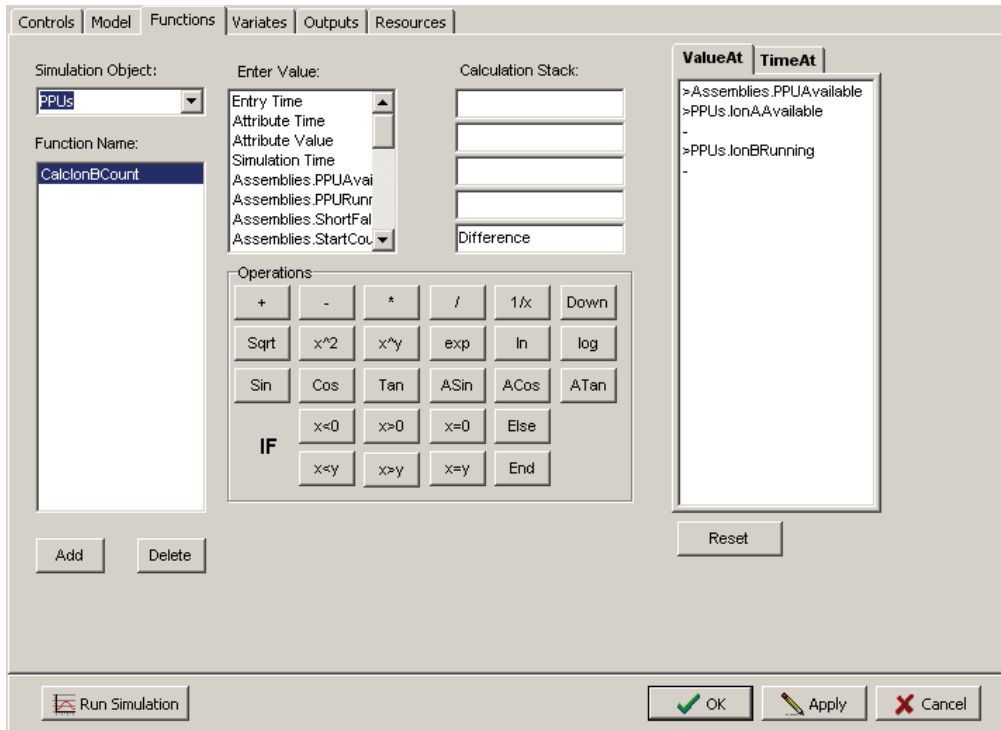
Figure 2.4: Simulation Model Tab Object View.



## Functions Tab

The “Functions” tab allows the analyst to define functions that can be used to set values of attributes (see next page). Each defined function actually defines two different functions: ValueAt which returns the attribute value possibly based on the current or future simulation time and TimeAt which returns a future simulation time when the attribute at the current rate will reach the entered value. The ValueAt function is used primarily in the state definition to set an attribute value at the current simulation time. The TimeAt function is used within the state transition definition where delay or duration times are needed. Thus an event can be placed in the simulation queue at the point in time when that value of the attribute is expected to equal the value given.

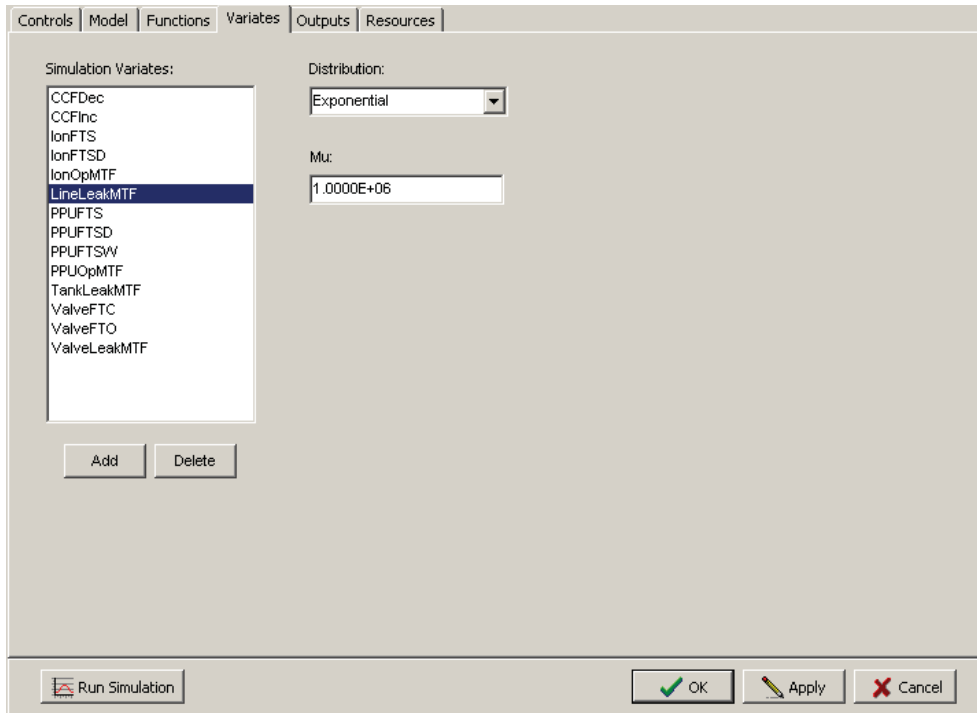
Functions are defined using a reverse Polish notation methodology where values are entered in a stack and then operators operate on the stack. Reverse Polish notation is a popular method on scientific calculators.



## Variates Tab

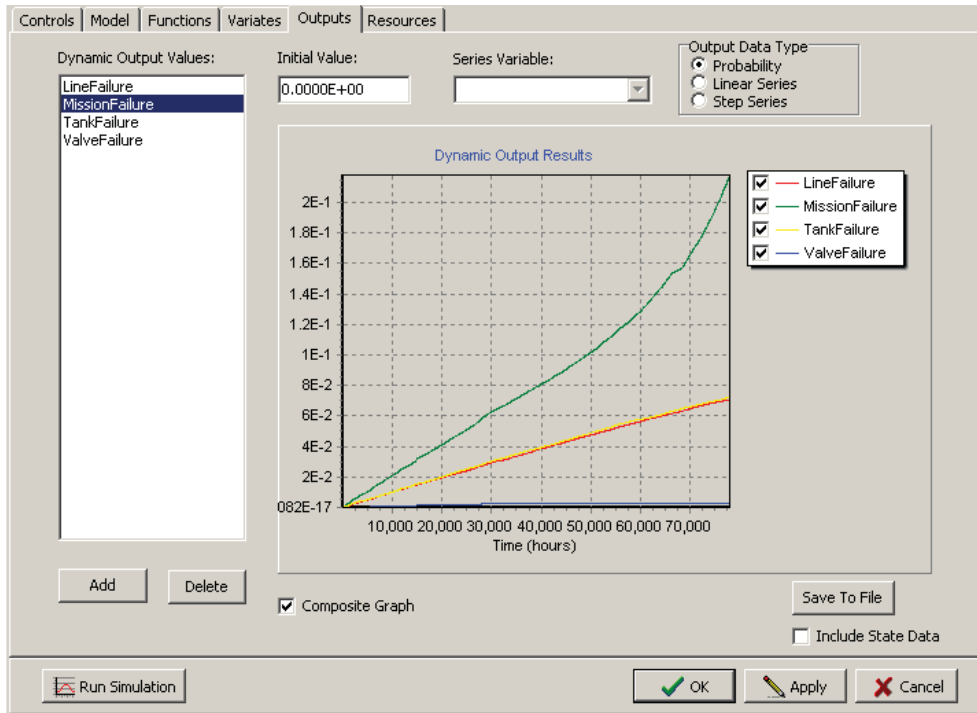
The “Variates” tab defines those different Variates that can be used throughout the simulation process (see next page). Variates provide random values based on a distribution and parameters of the distribution. The Variates are what gives the simulation its randomness and makes it useful for risk assessment.





## Outputs Tab

The “Outputs” tab controls the collection and display of simulation data (see next page) Each defined dynamic output stores a series of time-value pairs of data that describes the simulation run. The dynamic output is set to an initial value. It also defines an output type that includes probability which can record a single time of system failure and shows this data using a probability function plot. The series output types record the value of an attribute over time as an average value over the simulation time. This screen also gives the analyst a way of saving the data in a comma delimited file for use in other analysis tools.



One feature of the advance PRA approach is to use block diagrams to represent system behavior and transition states. The design described in this document will allow users to construct system depictions in terms of System Block Diagrams (SBD) where the nodes of the diagram will come from a library of typical components (e.g., nuclear power plant, electronic network, power distribution system, aerospace system). These diagrams will then, in turn, serve as the reliability model which can be either stand-alone (if working on a design specific to that system) or may be integrated into a larger model. Since tools like SAPHIRE has built-in modules for items like common-cause failure analysis, the block diagram module must automatically recognize when this failure mode is applicable, thereby freeing the system developer from needing to know highly specialized analysis techniques such as common-cause failure analysis.

## B-1 Object Information Library

A SBD is defined as a cyclic diagram consisting of nodes and arc. Nodes represent system objects such as physical components. Arcs represent the relationships between nodes and may represent causal mechanisms or physical flow depictions (e.g., fluid flow through a pumping system, network connections in an instrumentation system). A simple example of two SBDs is shown in B-1.



Figure B-1. Simple SBD examples, where A affects B and D affects C.



An advanced PRA project will allow the user to specify the general object information related to the SBD nodes. This information will include the information defined in Table 2.

The user will be able to view, modify, add, or delete the SBD information.

When a user **modifies** an object, this modified information will be used to update all diagrams that use that object. For example, if the three diagrams use a “globe value” object and the failure rate for that value is changed, all three diagrams would automatically use that updated information.

- When a user **adds** a new object, this information will be available in the object library.
-

Table 2. SBD object information.

Item	Data Structure	Description
Object name	24 characters	A general (short) name for the phase
Object description	120 characters	General descriptive text
Object 2D icon	Polygon shape and color definition	
Object 3D representation	OpenGL	
Object probability information	Basic Event	Probability information (failure rates, mission times, repair times, etc.) stored in the basic event relationship
Object category	Index key	Key to a user defined category identifying the first level of a hierarchy. For example, a category may be “valves.” Optional.
Object subcategory	Index key	Key to a user defined category identifying the second level of a hierarchy. For example, a subcategory may be “globe valves.” Optional.
Object positional information	3-D position, velocity, and time	<p>Positional information using the GPS Standard Positioning Service:</p> <p><a href="http://www.navcen.uscg.gov/gps/geninfo/">http://www.navcen.uscg.gov/gps/geninfo/</a></p> <p>Positional information should be stored using the Universal Transverse Mercator (UTM) Projection system.</p>

## B-2 Graphical SBD Editor

Block-type diagrams provided one of the fundamental building blocks to the analysis of complex systems in PRA. The definition of a reliability block diagrams is a model used to determine system reliability on a module or “block” basis rather than a component basis using a nodal diagram approach. For complex systems, the use of block diagrams makes the operations – and corresponding reliability – of a system much easier to understand than typical PRA methods, highlight system weaknesses much quicker since “pinch points” are evident from the model flow, and make what-if analyses much easier since nodes can be quickly added or removed from the model. As an example, we illustrate (Figure B-2) a simple diagram depicting an actual auxiliary feedwater system for a current generation pressurized water reactor. It is this type of figure that design engineers for next generation nuclear power plants understand, not the complex diagrams typical in a PRA.

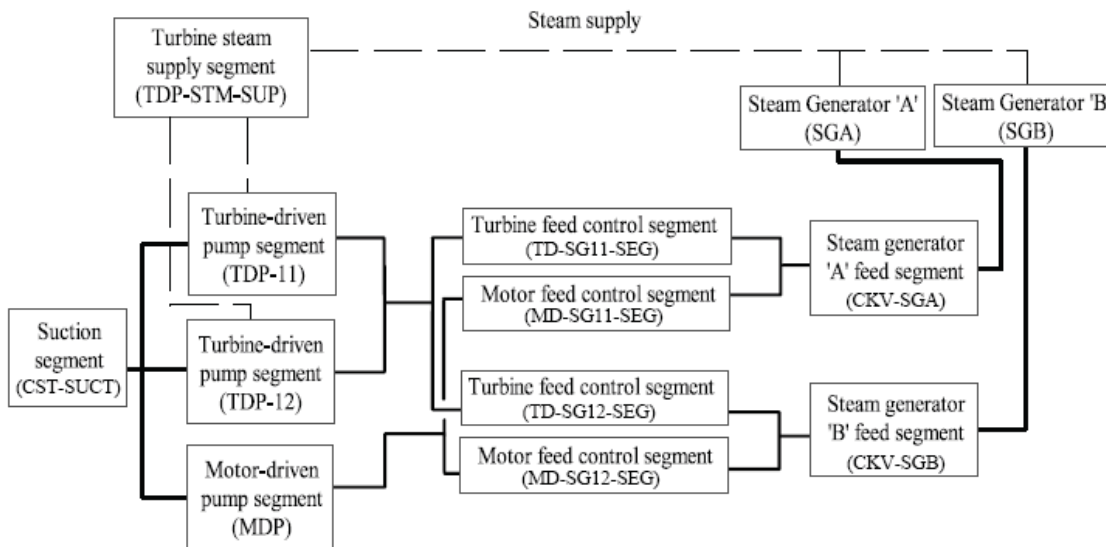


Figure B-2. A nuclear power plant auxiliary feedwater system recast as a block diagram.

The SBD editor will be a “drag-and-drop” tool to allow non-PRA engineers to construct models by connecting blocks, where blocks represents components or subsystems. The block diagram structure will be based upon object design, where the block diagram may consists of other block diagrams connected via arcs. By forcing this recursive type of design into the editor at the onset, the designer tool will allow nested levels of abstraction, thereby further allowing non-PRA specialist to construct models while at the same time providing a rich representation of the system operation. An example of the nested level of modeling is shown in Figure B-3.

Success criteria for subsystem component groups will be defined using the graphical editor and will be displayed as a part of the graphic.

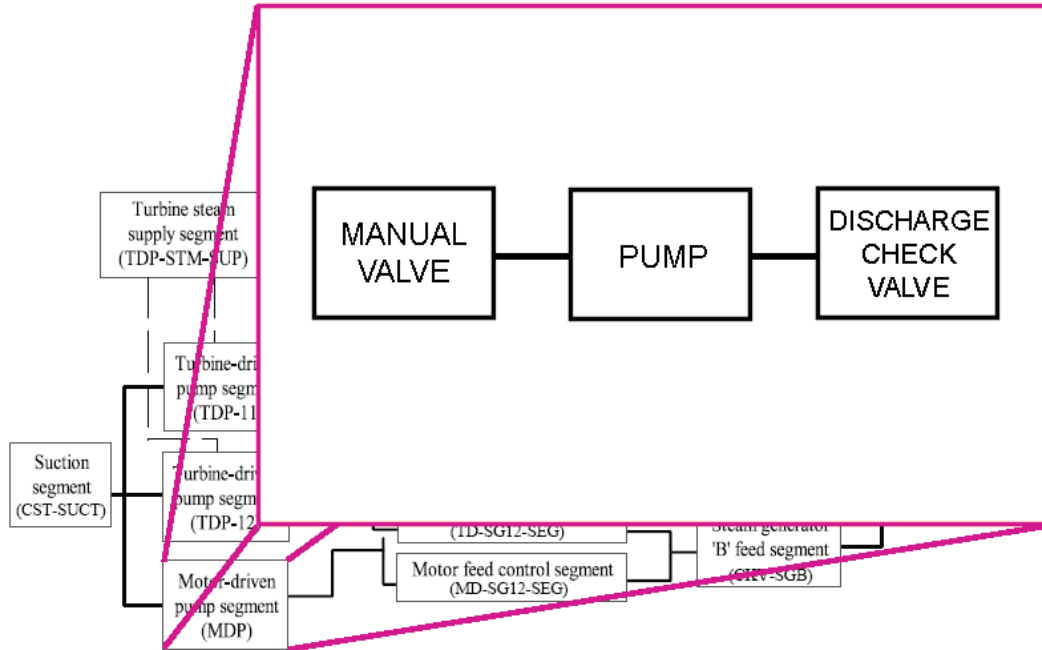


Figure B-3. An example of nesting block diagrams in order to construct complex models.

### B-3 SBD to Fault Tree Routine

Simple block diagrams may be turned directly into fault tree logic, which can be solved by the advanced PRA solution engine. For example, the motor-driven pump segment consists of three components – however this segment can be transmuted to an equivalent OR logic gate with the three component failures represented as inputs into the OR gate. Further, the three pump segments are connected in parallel, which will imply that redundancy exists for the system. Once the user indicates the success criteria (e.g., “I need two pumps to provide flow”), then the applicable logic structure can again be constructed and stored. In this manner, relatively complex models may be constructed simply by traversing through the parent-to-child relationships.

To perform the SBD to fault tree routine, we defined transformation rules for a series of cases. Examples of these cases are illustrated below in Figure B-4 and Figure B-5. More complex systems can be evaluated by recursively applying the rules described in these figures.

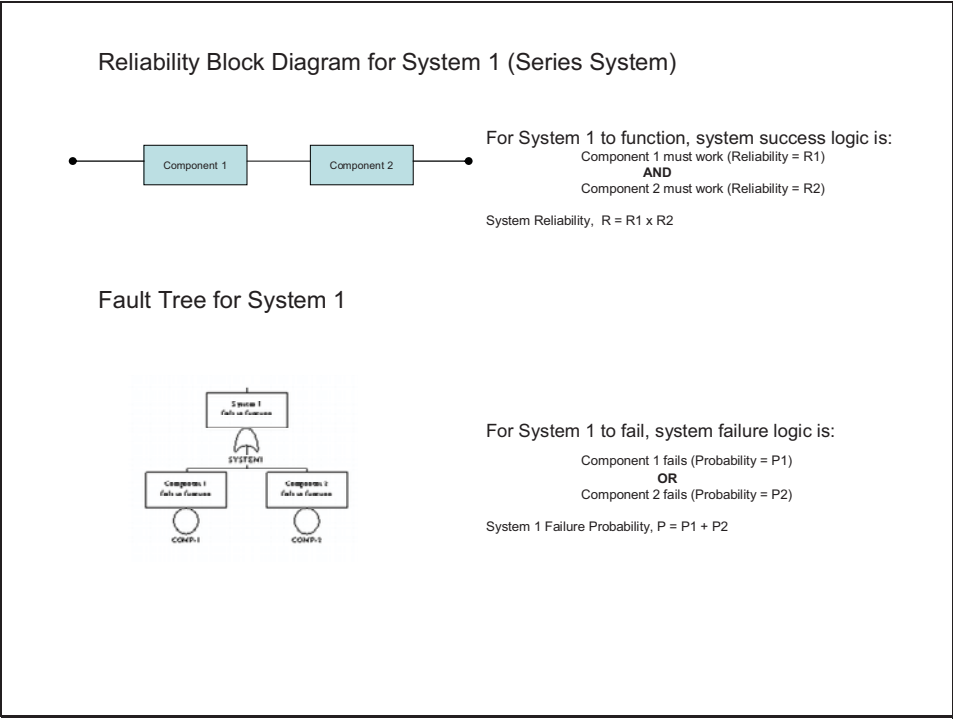


Figure B-4. SBD to fault tree rule for series components.

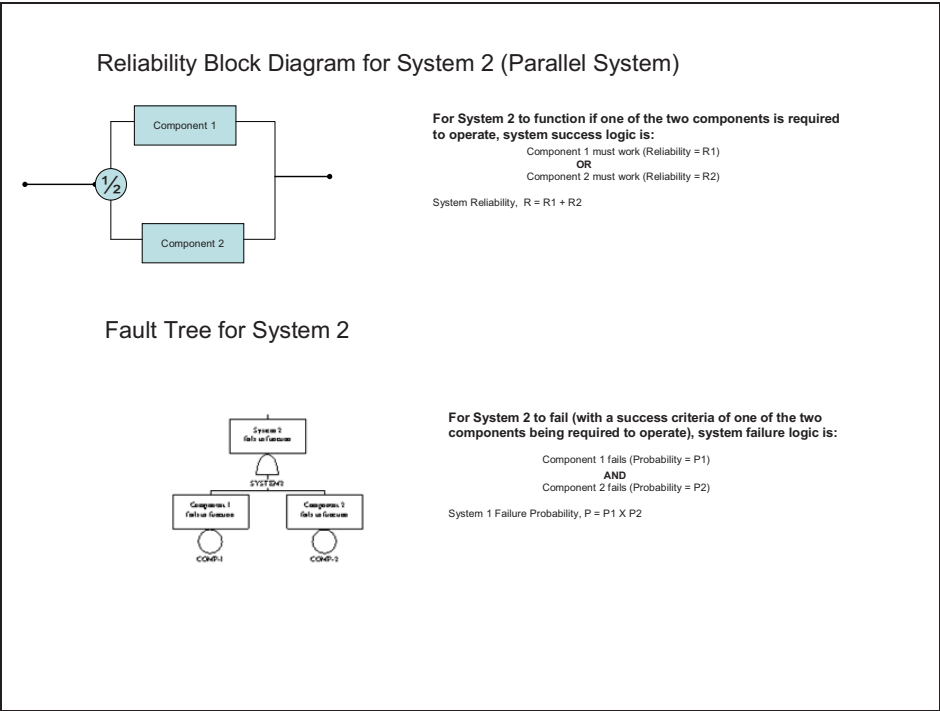


Figure B-5. SBD to fault tree rule for parallel components (1 of 2 must function).

The analysis of the numerical results is a pivotal element in PRA. The main scope is to take the information contained in the raw data<sup>13</sup> and project it into a form that allows the user to make decision.

This analysis is usually performed after the raw data is generated. However, we believe that this analysis should also be performed when the data is being generated. In order to perform the analysis of the data it is here suggested to move towards data mining techniques.

Data mining is a quite a generic term that includes a large spectrum of algorithms. As shown in Figure 26, these algorithms can be classified into four classes/applications [1]:

1. Cluster Analysis
2. Machine Learning
3. Manifold Analysis
4. Anomaly Detection

---

<sup>13</sup> Raw data contains information of:

- temporal profile of state variables (e.g., core temperature or containment pressure),
- timing of specific events, and,
- sequence of events



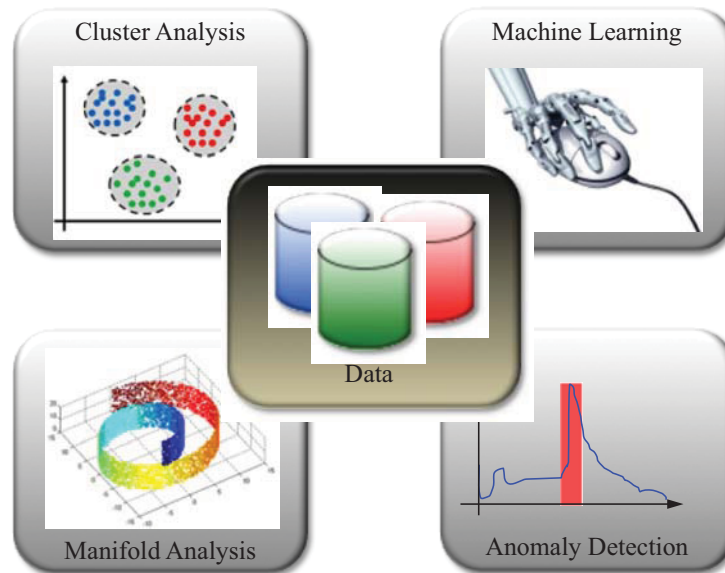


Figure 26. Overview of data mining.

These four categories and their application in advanced PRA are described in the next paragraphs.

### **Cluster Analysis**

Cluster analysis [3] is a generic term that refers to set of algorithms which performs analysis of the data by grouping them into clusters or classes. The analysis of these data is usually performed by:

- Considering a set of patterns;
- Grouping scenarios that have similar pattern (e.g., core damage or containment failure) into classes;
- Performing the analysis of each class separately.

The process called Classification is also known as supervised learning technique. It is still useful for organizing and analyzing large data sets, but has a major drawback: it requires the user to define the classes before the analysis. Clustering [5] (also known as unsupervised learning technique), on the other hand, aims to identify those classes automatically by looking at the geometric structure of the data.

In our application, the data comprise a set  $S$  of scenarios  $s_i$  ( $i=1, \dots, I$ ) where each scenario is characterized by its own probability  $p_i$  and by the temporal profile of the state variables  $x_j$  ( $j=1, \dots, J$ ) and system/component status  $c_k$  ( $k=1, \dots, K$ ) as functions of time. Clustering and classification algorithms aim to provide to the user with tools able to identify scenarios that have similar temporal behavior, group them, and find commonalities among them in terms of sequencing/timing of events as suggested in Figure 27.

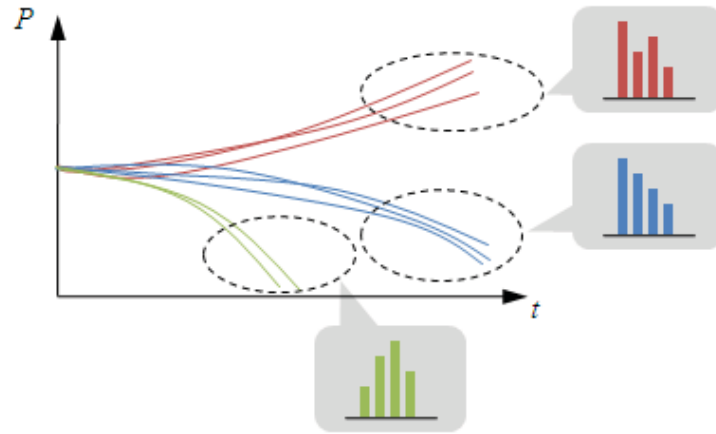


Figure 27: Scenario clustering: scenarios have been clustered in three clusters, and analysis is performed separately for each cluster.

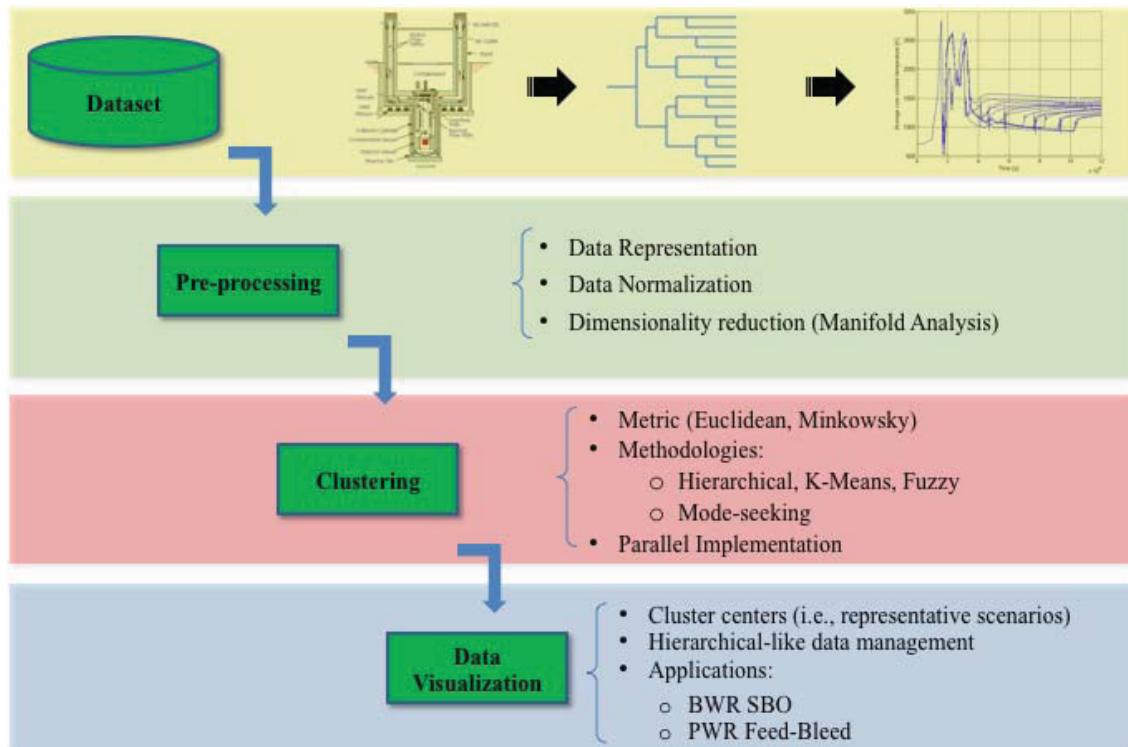


Figure 28: Data analysis: steps needed in the clustering process [10].

The structure of the clustering/classification process is the following [10] (see Figure 28):

1. The data being generated (e.g., from simulation) needs to be retrieved in a readable format (e.g., “.txt”). Data include temporal behavior of not just state variables but also components/systems status.
2. Before the clustering step, the raw data may need to be pre-processed. First, a set of state variables that is going to represent each scenario needs to be chosen. This process

can be performed by the user (i.e., he chooses the variables that he is interested in) or automatically. The last one can be performed using manifold analysis techniques based on Principal Component Analysis [6] or Multi-Dimensional Scaling algorithms. Due to the fact that the chosen variables may be different in both nature (e.g., temperature and pressure) and scale (i.e., the range of these variables might differ in terms of order of magnitude), it may be necessary to perform a scaling/normalization operation on the chosen variables.

3. Clustering/Classification is performed by starting from the metrics chosen by the user (e.g., Euclidean distance), which defines the structure of the data set (see Table 3 for a summary of the most used metrics). The best results have been performed using Mode-seeking algorithms (and shown in Figure 29), but classical algorithms, such as K-Means [9] or Fuzzy C-Means, can be developed as well. Since clustering algorithms can be time-consuming for large data sets, a parallel implementation of the algorithms is desirable.
4. Given the clusters obtained in the previous step, the goal is to provide the user with meaningful insight into the original data set. Further classification analysis may be required in order to gain such a meaningful insight (hierarchical-like data management).

Table 3: Summary of the commonly used distance based metrics.

Measure	Form
Minkowski distance	$d_n(\vec{x}, \vec{y}) = \left( \sum_{k=1}^{\delta}  x_k - y_k ^n \right)^{\frac{1}{n}}$
Euclidean distance	$d_2(\vec{x}, \vec{y}) = \left( \sum_{k=1}^{\delta}  x_k - y_k ^2 \right)^{\frac{1}{2}}$
Taxicab distance	$d_1(\vec{x}, \vec{y}) = \sum_{k=1}^{\delta}  x_k - y_k $
Supremum distance	$d_0(\vec{x}, \vec{y}) = \max_k  x_k - y_k $
Mahalanobis distance	$d_M(\vec{x}, \vec{y}) = (\vec{x} - \vec{y})^T S^{-1} (\vec{x} - \vec{y})$

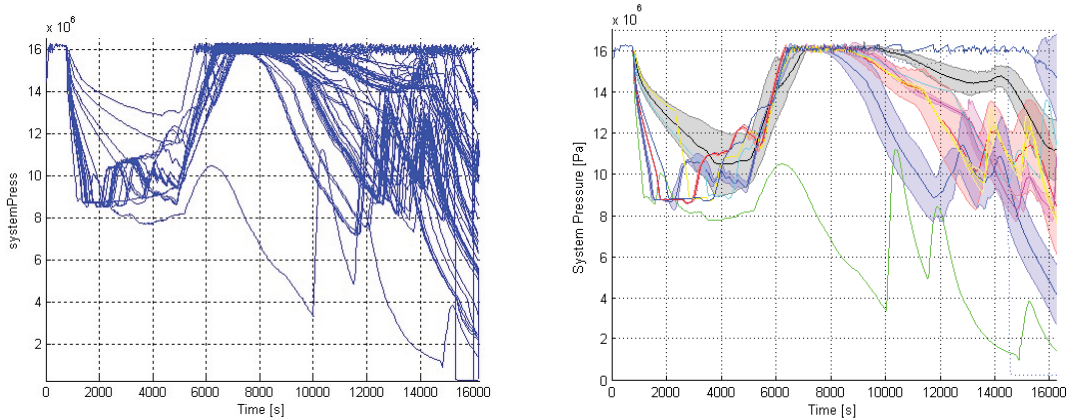


Figure 29: Example of scenario clustering [10].

### Survey of Classification Algorithms

From the literature it is possible to organize classification algorithms into the following five categories [8]:

- Artificial Intelligence based (Logical/Symbolic techniques): this category includes Decision trees and rule-based classifier.
- Perception based: this category includes single and multi layered perception (ANN), radial basis functions networks.
- Statistical based: this category includes Naïve Bayes classifiers and Bayesian networks.
- Instance based (e.g., kNN).
- Support Vector Machines (SVM).

Figure 30 [8] shows a comparison of some of the methodologies presented above.

### **Survey of Clustering Algorithms**

From the literature, it is possible to organize clustering methodologies into two classes based on their approach to the clustering problem:

- Hierarchical
- Partitional

Hierarchical algorithms organize data into a tree structure according to a proximity matrix in which each element  $(j, k)$  is some measure of the similarity (or distance) between the items to which row  $j$  and column  $k$  correspond. Usually, the final result of these algorithms is a binary tree, also called dendrogram, in which the root of the tree represents the whole data set and each leaf is a data point.

	Decision Trees	Neural Networks	Naïve Bayes	kNN	SVM	Rule-learners
Accuracy in general	**	***	*	**	****	**
Speed of learning with respect to number of attributes and the number of instances	***	*	****	****	*	**
Speed of classification	****	****	****	*	****	****
Tolerance to missing values	***	*	****	*	**	**
Tolerance to irrelevant attributes	***	*	**	**	****	**
Tolerance to redundant attributes	**	**	*	**	***	**
Tolerance to highly interdependent attributes (e.g. parity problems)	**	***	*	*	***	**
Dealing with discrete/binary/continuous attributes	****	***(not discrete)	***(not continuous)	***(not directly discrete)	** (not discrete)	***(not directly continuous)
Tolerance to noise	**	**	***	*	**	*
Dealing with danger of overfitting	**	*	***	***	**	**
Attempts for incremental learning	**	***	****	****	**	*
Explanation ability/transparency of knowledge/classifications	****	*	****	**	*	****
Model parameter handling	***	*	****	***	*	***

Figure 30: Comparing learning algorithms (\*\*\*\* stars represent the best and \* star the worst performance) [8].

Partitional clustering methodologies can be divided furthermore into five classes:

- *Squared error*: Squared error algorithms assign each point to the cluster whose center (also called centroid) is nearest. The center is the average of all the points in the cluster, that is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster. The most famous and used methodology is the K-Means algorithm [9].
- *Fuzzy*: Fuzzy clustering is very similar to K-Means but each point has a degree of belonging to every cluster, as in fuzzy logic, rather than belonging completely to just one cluster. Thus, points on the edge of a cluster may be in the cluster to a lesser degree than points in the center of cluster. For each point  $x_i$  there is a coefficient  $u_k(x_i)$  which gives the degree of being in the  $k^{\text{th}}$  cluster. An example of fuzzy clustering methodology is the Fuzzy C-Means algorithm.
- *Mode Seeking*: These methodologies are based on the assumption that the distribution of the points in the state space can be described through a probability density function. The goal is to find the modes, i.e., the regions in the state space with higher data densities. An example of this kind of methodology is the Mean-Shift methodology.
- *Graph Theoretical*: Graph theory based methodologies aim to build a graph of the data set often called Minimal Spanning Tree. Clusters are determined by deleting the longest edges of the graph. Conceptually this approach is very similar to the hierarchical one.

- *Neural Network*: Neural network methodologies are essentially inspired by the biological neural network. The learning process associated with the training of an artificial neural network (ANN) allows to associate patterns (input variables) to clusters (output nodes) through a series of weights that are updated at each iteration.

## **Machine Learning**

Machine learning is a discipline geared toward the development of algorithms able to learn from data generated in the past and foresee the future.

In PRA applications, as an example, we are interested into the evaluation of the conditions under which failure occurs. In particular, we are looking for the limit surface: the conditions that separate simulations leading to desired outcomes (i.e., core intact) from ones leading to undesired outcomes and (i.e., core damage).

Starting from an initial set of samples (i.e., a training set) the algorithm, at each iteration, performs the following steps:

- 1) Evaluate the limit surface
- 2) Choose the coordinates in the Issue Space of the next sample (using sampling algorithms);
- 3) Generate the input for the system simulator
- 4) Perform the simulation given the sample coordinate determined in step 2
- 5) Store the information of the simulation into a database
- 6) Return to step 1.

This process stops when a user-specified objective function is reached (e.g., the limit surface just obtained does not differ from the one obtained in the previous iteration). Machine-learning algorithms that can perform this task are:

- Support vector machines (SVM) [21]
- Neural networks (ANN) [12]
- Genetic algorithms (GA)
- Hidden Markov Models (HMM) [16]

In order to decrease the computational time of the analysis, low fidelity simulation models (i.e., less computationally intensive) can be used in the initial stage of the analysis. When the limit surface is approximating the real one, high fidelity models (i.e., more computationally intensive) can be used to refine the objective function.

### **Impact on Advanced PRA**

Improvements for Advanced PRA would be:

- Smart sampling of simulations
- Computational resources allocation efficiency
- Decrease of computational time

### **Manifold Analysis**

When performing data analysis, data dimensionality reduction techniques are usually employed in order to decrease computational time. Two classes of data dimensionality reduction techniques exist:

- Data cardinality reduction

- Data dimensionality reduction or Manifold analysis

While the algorithms in the first class aim to reduce the number of data points of the original data set (e.g., clustering algorithms), the ones in the second class aim to identify correlation among data dimensions and reduce the redundant ones.

In our applications, due to the large complexity of the system, the dimensionality (i.e., the information contained in each scenario) is very large. A generic NPP dynamic system model (e.g., modeled using RELAP-5) is composed by thousands of nodes and each node is characterized by a set of state variables (e.g., node temperature, pressure). These variables are locally high correlated (e.g., due to conservation laws), but this correlation fades for variables of distant nodes.

In this case, dimensionality reduction algorithms aim to reduce the amount of redundant information (caused by locally high correlations). In the literature, it is possible to find the following dimensionality reduction algorithms:

- Principal Component Analysis (PCA) [6]
- Multi-dimensional Scaling (MDS)
- Principal Curves [2]
- Kernel PCA [18]
- ISOMAP [22]
- Local Linear Embedding (LLE) [17]

### **Impact on Advanced PRA**

Improvements for Advanced PRA would be:

- Generation of surrogate models
- Decrease in computational time during data clustering
- Identification of variable correlations

### **Anomaly Detection**

Thermo-hydraulic codes have a limited range of applicability due to the fact that correlations and models implemented are defined over a fixed domain. When the simulation crosses this range of applicability, the numerical errors can greatly bias the numerical results. These numerical errors generate artifacts, i.e. anomalies of the temporal behavior of the solution.

Anomaly detection algorithms aim to identify these anomalies while the simulation is still running. When an anomaly is found, the simulation is stopped and left in stand-by waiting for user verification. An example of anomaly detection algorithm would be based on clustering algorithms: anomalies would be considered as outliers (i.e., points located far from an initial set of simulations).

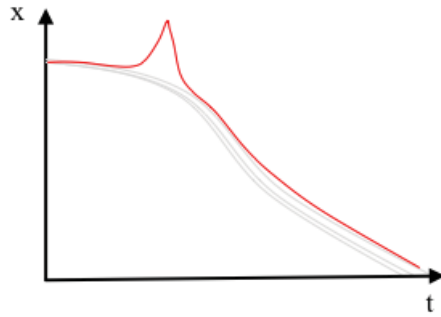


Figure 31: Example of anomaly (scenario in red) in scenario generation.

### **Impact on Advanced PRA**

Improvements for Advanced PRA are:

- Reduction in simulation time
- Efficiency of the calculation process
- Verification of the PRA results

### **Tools available**

The scope of this section is to give a brief overview of the data mining and visualization tools that are available and that can be embedded into Advanced PRA. These tools are listed in Table 4.



Table 4: Tools available for Data Mining

Name	Type	Language	Capabilities	Notes
R	Environment	R and C	Linear and non linear statistical analysis Advanced clustering Advanced classification Graphics visualization tools	Open Source
OpenBugs	Environment	Pascal	Bayesian analysis Linear and non linear statistical analysis	Open Source
GNU Octave	Environment	C++		
Dakota	Library	C++	Linear and non linear statistical analysis Uncertainty analysis Parallel computing	
matplotlib	Library	Python	Graphics Visualization tools	Open Source
JHepWork		Java	Graphics visualization tools	Open Source
SciPy	Library	Python	Linear and non linear statistical analysis Clustering Classification	Open Source
Gnuplot	Command-line	C	Graphics visualization tools	Open Source
ggplot2	Library	R	Graphics visualization tools	Open Source

### References for Appendix C

- [1] Han, J., Kamber, M., Pei, J., *Data Mining: Concepts and Techniques*, 3rd ed., Morgan Kaufmann (2011).
- [2] Hastie, T., Stuetzle, W., Principal curves, *Journal American Statistical Assoc.*, vol. 84: 502-516, (1989).
- [3] Hastie, T., Tibshirani, R., Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2nd edition.
- [4] Ingman, D., Reznik, L.A., "Dynamic character of failure state in damage accumulation processes", *Nuclear Science and Engineering*, vol. 107: 284-290 (1991).
- [5] Jain, K., Murty, M., Flynn, P., "Data clustering: A review," *ACM Comput. Surv.*, vol. 31: 264-323 (1999).
- [6] Jolliffe, I.T., *Principal Component Analysis*, Springer-Verlag (1986).

- [7] Kaplan, S., Garrick, B.J., "On the quantitative definition of risk", *Risk Analysis*, vol. 1, 11-28 (1981).
- [8] Kotsiantis, S.B., "Supervised Machine Learning: A Review of Classification Techniques", *Informatica*, vol. 31, 249-268 (2007).
- [9] MacQueen, J. B., "Some methods for classification and analysis of multivariate observations," in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* (L. M. L. Cam and J. Neyman, eds.), vol. 1: 281-297, University of California Press (1967).
- [10] Mandelli D., "Scenario Clustering and Dynamic Probabilistic Risk Assessment", Doctorate dissertation, The Ohio state University (2011).
- [11] Mandelli, D., Aldemir, T., Yilmaz, A., Metzroth, K., Denning, K., "Scenario aggregation in dynamic probabilistic risk assessment," Draft for *Reliability Engineering and System Safety*, (2012).
- [12] Mandic, C., Chamber, J., *Recurrent Neural Networks for Prediction: Architectures, Learning algorithms and Stability*, Wiley (2001).
- [13] Marseguerra, M., Zio, E., "Monte Carlo approach to PSA for dynamic process systems", *Reliability Engineering and System Safety*, vol. 52(3): 227–241 (1996).
- [14] Metzroth, K., "A Comparison of Dynamic and Classical Event Tree Analysis for Nuclear Power Plant Probabilistic Safety/Risk Assessment", Doctorate dissertation, The Ohio state University (2011).
- [15] Nelson, B.L., *Stochastic modeling: analysis and simulation*, Dover (1995).
- [16] Rabiner, L.R., "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE*, vol. 77(2): 257-285 (1989).
- [17] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290: 2323-2328 (2000).
- [18] Scholkopf, B., Smola, A., Muller, K., "Nonlinear component analysis as a kernel eigenvalue problem", *Neural Comp.*, Vol. 10(5): 1299-1319 (1998).
- [19] Sharma, A. K., Sheikh, S., Pelczer, I., Levy, G., "Classification and clustering using neural networks," *Journal of Chemical Information and Computer Sciences*, vol. 34(5): 1130-1139 (1994).
- [20] Siu, N., "Risk assessment for dynamic systems: an overview", *Reliability Engineering and System Safety*, vol. 43(1): 43-73 (1994).
- [21] Steinwart, G., and Christmann, A., *Support Vector Machines*. Springer-Verlag, New York (2008).
- [22] Tenenbaum, J. B., de Silva, V., Langford, J. C., "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290: 2319-2323 (2000).
- [23] United State Nuclear Regulatory Commission. *NUREG-1150, Severe Accident Risks: An Assessment for Five U.S. Nuclear Power Plants*, United State Nuclear Regulatory Commission Washington, D.C. (1990).
- [24] Zhan, C. T., "Graph theoretical methods for detecting and describing gestatlt clusters," *IEEE Transaction on Computers*, vol. 1(20): 68-86 (1971).