INL/RPT-23-74447

Light Water Reactor Sustainability Program

Methods and Feature Enhancements for Industry Use of EMRALD

September 2023

U.S. Department of Energy

Office of Nuclear Energy

INL/RPT-23-74447

Methods and Feature Enhancements for Industry Use of EMRALD

Prianka Pandit
Steven Prescott
Daniel Nevius
Mihai Diaconeasa

September 2023

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

ABSTRACT

Event Modeling Risk Assessment using Linked Diagrams (EMRALD) is a dynamic probabilistic risk assessment (PRA) tool developed by the Idaho National Laboratory. It is an open-source tool that has been used for many research projects along with more recent academic and industry projects. To expand for more industry use and analysis, a strong validation framework and test cases needed to be developed. Also, additional features needed to be added to enable some scenarios and ease of use. This report goes over the initial validation framework setup and testcase as well as the software modification from this project. Initially, this project was going to include implementing methods for coupling with classical PRA tools. However, earlier research concluded that coupling methods would only be mathematically valid for limited scenarios, so work was pivoted to other areas of improving industry use.

Page intentionally left blank

CONTENTS

ABSTRACT iii

ACRONYMS x

1. BACKGROUND OF EMRALD FOR DPRA 1
   - 1.1 Development History 1
   - 1.2 Lab Research, Industry, and Academic Use 1
   - 1.3 Classical and Dynamic PRA Coupling & Pivot 2

2. EMRALD VALIDATION WORK 2
   - 2.1 Validation Framework 2
   - 2.2 Dynamic vs. Static Evaluation Cases 3
     - 2.2.1 Single Component Failure 4
     - 2.2.2 Single Component Failure and Repair 5
     - 2.2.3 Two Identical Components in Parallel Fail 6
     - 2.2.4 Two Identical Components in Series Fail 8
     - 2.2.5 Two Identical Components in Parallel Fail with Common Cause Failure 9
     - 2.2.6 Initiating Event with One Engineering Safety Feature 10
     - 2.2.7 Initiating Event with Engineering Safety Feature 1 12
     - 2.2.8 Initiating Event with Engineering Safety Feature 2 13
   - 2.3 Dynamic Model Numerical Evaluation Cases 14
     - 2.3.1 Two Identical Components in Active Parallel Fail and Get Repaired, One Repairperson Available 16
     - 2.3.2 Two Identical Components in Active Parallel Fail and Get Repaired, Two Repairpersons Available 19
     - 2.3.3 Two Identical Components, One Active and One in Standby, Fail and Get Repaired, One Repairperson Available 21
     - 2.3.4 Two Identical Components, One Active and One in Standby, Fail and Get Repaired, Two Repairpersons Available 23
     - 2.3.5 Two Identical Components in Series Fail and Get Repaired, One Repairperson Available 25
     - 2.3.6 Two Identical Components in Series Fail and Get Repaired, Two Repairpersons Available 27

3. EMRALD ENHANCEMENTS 30
   - 3.1 Solve Engine 30
   - 3.2 Modeling User Interface 30
   - 3.3 Future Updates 30

4. EMRALD RESULTS AND VISUALIZATION 30
   - 4.1 Standard Result Capabilities 31
   - 4.2 Graphical Base Results 31
     - 4.2.1 Path Results Data 31
     - 4.2.2 Time-Based Sankey 32
     - 4.2.3 GUI Development 33

5. REFERENCES 34

Appendix A 36

FIGURES

Figure 1. An example of an EMRALD system test. 3

Figure 2. Fault tree in SAPHIRE for single component failure. 4

Figure 3. EMRALD model for failure of a single component. 5

Figure 4. Markov model for single component failure and repair. 5

Figure 5. SAPHIRE model for the failure and repair of a single component. 6

Figure 6. EMRALD model for the failure and repair of a single component. 6

Figure 7. EMRALD model for the failure of two identical components in parallel. 7

Figure 8. SAPHIRE model for the failure of two identical components in parallel. 7

Figure 9. EMRALD model for the failure of two identical components in series. 8

Figure 10. SAPHIRE model for the failure of two identical components in series. 9

Figure 11. EMRALD model for the failure and repair of two components in series, one repairperson available. 10

Figure 12. EMRALD model of an event tree with one component as the engineering safety feature. 11

Figure 13. SAPHIRE model of an event tree with one component as the engineering safety feature. 11

Figure 14. EMRALD model of an event tree with two components in parallel as the engineering safety feature. 12

Figure 15. SAPHIRE model of an event tree with two components in parallel as the engineering safety feature. 12

Figure 16. EMRALD model of an event tree with two components in series as the engineering safety feature. 13

Figure 17. SAPHIRE model of an event tree with two components in series as the engineering safety feature. 14

Figure 18. Markov model for failure of two components in parallel, one repairperson available. 17

Figure 19. EMRALD model for the failure and repair of two components in parallel, one repairperson available. 18

Figure 20. EMRALD model of the Markov model for the failure and repair of two components in parallel, one repairperson available. 18

Figure 21. Markov model for failure of two components in parallel, two repairpersons available. 19

Figure 22. EMRALD model for the failure and repair of two parallel components, two repairpersons available. 20

Figure 23. EMRALD model of the Markov model for the failure and repair of two parallel components, two repairpersons available. 20
Figure 24. SAPHIRE model for failure of two identical components in parallel, which can be repaired simultaneously. 20
Figure 25. Markov model for failure of two components, one active and one in standby, one repairperson available. 21
Figure 26. EMRALD model for the failure and repair of one active and one standby component, one repairperson available. 22
Figure 27. EMRALD model of Markov model for the failure and repair of one active and one standby component, one repairperson available. 22
Figure 28. Markov model for failure of two components, one active and one in standby, two repairpersons available. 23
Figure 29. EMRALD model for the failure and repair of one active and one standby component, two repairpersons available. 24
Figure 30. EMRALD model of Markov model for the failure and repair of one active and one standby component, two repairpersons available. 25
Figure 31. Markov model for the failure and repair of two components in series, one repairperson available. 26
Figure 32. EMRALD model for the failure and repair of two components in series, one repairperson available. 27
Figure 33. EMRALD model of Markov model for the failure and repair of two components in series, one repairperson available. 27
Figure 34. Markov model for the failure and repair of two components in series, one repairperson available. 28
Figure 35. EMRALD model for the failure and repair of two components in series, one repairperson available. 29
Figure 36. EMRALD model of Markov model for the failure and repair of two components in series, one repairperson available. 29
Figure 37. SAPHIRE model for the failure of two identical components in series, which can be repaired one a time. 29
Figure 38. Example of EMRALD UI with results. 31
Figure 39. Simple Sankey result diagram with a hover over to show statistics. 32
Figure 40. Time-based Sankey diagram showing the time and standard deviation whiskers. 32
Figure 41. Sankey results from a complicated model. 33
Figure 42. Option to open the Sankey results from the EMRALD solve engine GUI. 33
Figure 43. Options at the top of the Sankey result viewer. 33

TABLES

Table 1. Summary of dynamic vs. static evaluation cases. 3
Table 2. Data for validating the failure of a single component. 5
Table 3. Data for validating the failure of a single component. 6
Table 4. Data for validating the failure of two identical components in parallel. 8
Table 5. Data for validating the failure of two identical components in series. 9
Table 6. Data for validating the failure and repair of a two components in series, repaired one at a time. 10
Table 7. Data for validation of failure frequency of an event tree with one component as the engineering safety feature. 12
Table 8. Data for validating the failure frequency of an event tree with two components in parallel as the engineering safety feature. 13
Table 9. Data for validating the failure frequency of an event tree with two components in series as the engineering safety feature. 14
Table 10. Summary of dynamic vs. static evaluation cases. 14
Table 11. Data for validating the failure and repair of two components in parallel, repaired one at a time. 18
Table 12. Data for validating the failure and repair of two components in parallel, repaired simultaneously. 21
Table 13. Data for validating the failure and repair of two components, one active and one in standby, repaired one at a time. 23
Table 14. Data for validating the failure and repair of two components, one active and one in standby, repaired simultaneously. 25
Table 15. Data for validating the failure and repair of two components in series, repaired one at a time. 27
Table 16. Data for validating the failure and repair of two components in series, repaired one at a time. 29

Page intentionally left blank

ACRONYMS

CAFTA	Computer Aided Fault Tree Analysis
DPRA	Dynamic Probabilistic Risk Assessment
EMRALD	Event Modeling Risk Assessment Using Linked Diagrams
FY	Fiscal Year
GUI	Graphical User Interface
HRA	Human Reliability Analysis
HUNTER	Human Unimodel for Nuclear Technology to Enhance Reliability
INL	Idaho National Laboratory
JSON	JavaScript Object Notation
LDRD	Laboratory Directed Research and Development
LWRS	Light Water Reactor Sustainability
MeV	Modeling, Experimentation, and Validation
MTTF	Mean Time To Failure
NPP	Nuclear Power Plant
NRC	Nuclear Regulatory Commission
PRA	Probabilistic Risk Assessment
SAPHIRE	Systems Analysis Program for Hands-on Integrated Reliability Evaluations
SBIR	Small Business Innovation Research
SVG	Scalable Vector Graphics
TCF	Technology Commercialization Fund
UI	User Interface

METHODS AND FEATURE ENHANCEMENTS FOR INDUSTRY USE OF EMRALD

1. BACKGROUND OF EMRALD FOR DPRA

Event Modeling Risk Assessment using Linked Diagrams (EMRALD) is a Dynamic Probabilistic Risk Assessment (DPRA) tool developed at Idaho National Laboratory (INL) and has been used for many lab and academic research projects along with commercial endeavors. To expand for more industry use, continued feedback from users is used for development. Based on feedback, validation methods have been added. This report goes over a brief history of EMRALD and some of the latest features added that make it applicable for industry use.

1.1 Development History

EMRALD started out as a Laboratory Directed Research and Development (LDRD) project in 2005 and 2006 called "Dynamic Probabilistic Extensions to the Systems Analysis Program for Hands-on Integrated Reliability Evaluations (SAPHIRE) Risk and Reliability Designer Tool." In 2016, the Light Water Reactor Sustainability (LWRS) program needed a way to link external event simulations dynamically to risk analysis with the timing of component failures and operator actions. EMRALD was further developed to be used for external hazard analysis research projects [1][2][3][4]. The Nuclear Regulatory Commission (NRC) also looked at EMRALD and a flooding simulation software called Neutrino for performing a case study of flooding at a nuclear power plant (NPP) [5]. EMRALD was then selected for an Energy I-Corps Cohort 5 program to help engage with industry and find market areas for the software [6]. Through a phase I and II Technology Commercialization Fund (TCF) and a Small Business Innovation Research (SBIR) with FPoliSolutions, EMRALD achieved some critical development features and was open sourced to be made available to the public [7][8]. Other minor modifications have been made for specific research or industry needs.

1.2 Lab Research, Industry, and Academic Use

The initial use of EMRALD for INL research was for dynamic flooding analysis. This was expanded to look at multi-hazard events such as seismic-induced flooding. With the maturing of the software after going open source, EMRALD was looked at for other analysis projects where timing was a critical feature. EMRALD has been a key tool over the last 3 years in the LWRS Physical Security Pathway. It is used to couple force-on-force simulations with thermal hydraulics while providing the operator actions and probabilistic risk assessment (PRA) as part of the model. This physical security work is key for evaluating if alternative protective strategies are numerically equivalent in replacing guard posts and reducing costs [9][10]. There has also been research projects looking at human reliability analysis (HRA) and using EMRALD for modeling operator actions [11][12]. Ongoing work has also added a Human Unimodel for Nuclear Technology to Enhance Reliability (HUNTER) HRA module to EMRALD to make it easier to include HRA conditions when modeling [13].

Since EMRALD is open source, INL does not always know about all the projects using EMRALD. INL has received several inquiries from universities and students to use EMRALD and has worked with them on several projects [14][15][16]. EMRALD training at the Modeling, Experimentation, and Validation (MeV) school was also provided where students looked at projects that could use the various tools they learned about [17].

INL has worked with three industry collaborators on coupling EMRALD with their software. First, FPoliSolutions on their integrated risk-informed modeling suite [18]. Second, with ARES Security on their Avert/EMRALD coupled force-on-force simulation software [19]. Finally, with Centroid Lab and linking with their Neutrino flooding simulation software [20]. Inquiries have also been received from other domestic and foreign businesses.

1.3 Classical and Dynamic PRA Coupling & Pivot

Earlier research from fiscal year (FY) 2022 into FY 2023 looked at methods for coupling static/classical PRA and dynamic PRA. The intent was to expand or simplify while taking advantage of some classical PRA benefits by coupling EMRALD directly with existing tools such as SAPHIRE and Computer Aided Fault Tree Analysis (CAFTA) [21]. This research looked at several possible methods of coupling or using the two PRA tools together. In the end, only one method that adjusted results was mathematically sound. However, it would only be useful for select scenarios. With these results, the continued work was pivoted to other identified tasks to improve industry use, namely validation and software enhancements discussed in the next two sections.

2. EMRALD VALIDATION WORK

Validation is key for scientific software, especially in the nuclear industry. This section goes over the built-in validation framework and initial validation cases recently added to EMRALD.

2.1 Validation Framework

EMRALD has a testing project that is part of the open source code. This project started out as unit testing and system testing. To simplify making and evaluating tests, a framework was constructed. This framework consists of several pieces:

- Data folders – All the models for testing are stored in a specified test folder. Any files needed for results comparison are also stored in a specified folder.
- Common test functions – Common functions are called at the beginning of a test. These functions create or clear a temporary storage location, copy a model test file, and compare test results.
- Execution module – For tests that run an EMRALD model, such as system tests or the validation models in Sections 2.2 and 2.3, specific simulation parameters are assigned to the execution module, and then it is run.

This framework standardizes how tests are made and minimizes the number of lines of code for each as shown in Figure 1. The test maker copies an existing test or template, follows a few common steps, and verifies the test is correct before adding it to the source repository.

```
[Fact]
| 0 references | pressr-inl, 180 days ago | 1 author, 1 change | 1 work item
public void DistEvent_LogNormal()
{
  string testName = GetCurrentMethodName(); //function name must match the name of the test model

  //Setup directory for unit test
  string dir = SetupTestDir(testName);
  //initial options, and optional results to save/test
  JObject optionsJ = SetupJSON(dir, testName, true);

  //Change the default settings as needed for the test seed default set to 0 for testing.
  optionsJ["inpfile"] = MainTestDir() + ModelFolder() + testName + ".json";
  optionsJ["runct"] = 100000;
  JSONRun testRun = new JSONRun(optionsJ.ToString());
  Assert.True(TestRunSim(testRun));

  //Uncomment to update the validation files after they verified correct
  //CopyToValidated(dir, testName, optionsJ);

  //compare the test result and optionally the paths and json if assigned
  Compare(dir, testName, optionsJ);
}
```

Figure 1. An example of an EMRALD system test.

A new block for user validation tests was added to the testing project and the EMRALD documentation. The goal is to provide validation documentation that goes along with the automated testing suite. This can be used as part of the software quality assurance for EMRALD.

2.2 Dynamic vs. Static Evaluation Cases

Part of the validation was to verify that the simulation-based results are equivalent to static PRA calculations using SAPHIRE and demonstrate the modeling in EMRALD. The following subsections go over the test cases performed where SAPHIRE was used along with analytical calculations to validate EMRALD results. The results of these validation test cases are summarized in Table 1 and denote the failure rate of a component as λ, the repair rate as μ, the availability as $A(t)$, the mission time as t, and $P(f)$ as the failure probability.

Table 1. Summary of dynamic vs. static evaluation cases.

Model #	Model Name	Test #	SAPHIRE Results	EMRALD Results
2.2.1	Single Component Failure	1	P(f) = 1×10^{0}	P(f) = 9.9995×10^{-1}
		2	P(f) = 6.3129×10^{-1}	P(f) = 6.3129×10^{-1}
		3	MTTF = 364.11 days	MTTF = 365 days
2.2.2	Single Component Failure and Repair	1	A(t) = 1.88172×10^{-2}	A(t) = 1.888×10^{-2}
		2	P(f) = 2.733×10^{-3}	P(f) = 2.84×10^{-3}
2.2.3	Two Identical Components in Parallel Fail	1	P(f) = 9.9995×10^{-1}	P(f) = 1×10^{0}
		2	P(f) = 3.9728×10^{-1}	P(f) = 3.9728×10^{-1}

Model #	Model Name	Test #	SAPHIRE Results	EMRALD Results
		3	MTTF = 547 days	MTTF = 547.5 days
2.2.4	Two Identical Components in Series Fail	1	MTTF = 5 days	MTTF = 5 days
		2	P(f) = 8.638×10^{-1}	P(f) = 8.638×10^{-1}
		3	MTTF = 178 days	MTTF = 182.5 days
2.2.5	Two Identical Components in Parallel Fail with Common Cause Failure	1	P(f) = 0.5861 (Beta Factor Model)	P(f) = 0.7461
2.2.6	Initiating Event	1	$f_0 = 0.005/year$	$f_0 = 0.00494/year$
2.2.7	Initiating Event with One Engineering Safety Feature	1	$f_{CD} = 3.160 \times 10^{-3}$	$f_{CD} = 3.101 \times 10^{-3}$
2.2.8	Initiating Event with Engineering Safety Feature 1	1	$f_{CD} = 2.008 \times 10^{-3}$	$f_{CD} = 1.986 \times 10^{-3}$
2.2.9	Initiating Event with Engineering Safety Feature 2	1	$f_{CD} = 4.32 \times 10^{-3}$	$f_{CD} = 4.51 \times 10^{-3}$

2.2.1 Single Component Failure

We can test the failure probability and mean time to failure (MTTF) of a single component given the mission time and failure rate. The SAPHIRE model and EMRALD model for the system are given in Figure 2 and Figure 3. The equation to calculate the MTTF is given in (1). We use a range of failure rates and mission times as shown in Table 2.

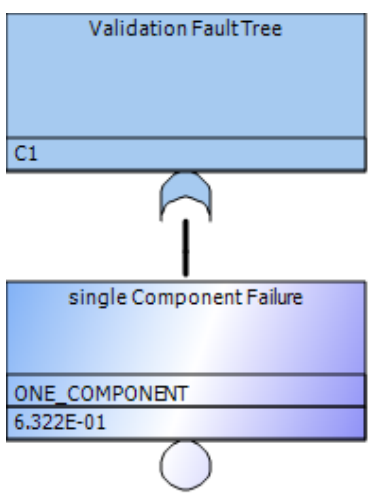


Figure 2. Fault tree in SAPHIRE for single component failure.

$$MTTF = \frac{1}{\lambda} \tag{1}$$

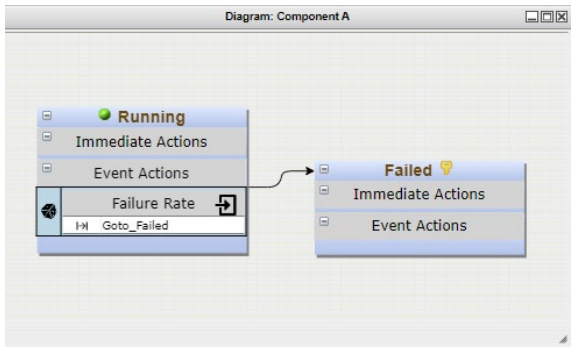


Figure 3. EMRALD model for failure of a single component.

Table 2. Data for validating the failure of a single component.

Test Number	Failure Rate	Mission Time	Number of Runs	EMRALD Failure Probability	SAPHIRE Failure Probability	EMRALD MTTF	Analytical MTTF
1	$\frac{1}{10\ days}$	100 days	100,000	$P(f) = 9.9995 \times 10^{-1}$	$P(f) = 1 \times 10^{0}$	MTTF = 10 days	MTTF = 10 days
2	$\frac{1}{365\ days}$	365 days	100,000	$P(f) = 6.3129 \times 10^{-1}$	$P(f) = 6.3129 \times 10^{-1}$	-	-
3	$\frac{1}{365\ days}$	10000 days	100,000	$P(f) = 1 \times 10^{0}$	$P(f) = 1 \times 10^{0}$	MTTF = 364.11 days	MTTF = 365 days

The MTTF is not compared in Test 2 because the mission time is not large enough to simulate enough failures to get an accurate result. The mission time was increased in Test 3 and compare the MTTF.

2.2.2 Single Component Failure and Repair

We can test the failure and repair of a single component given the failure rate, repair rate or time, and mission time. Using the repair rate, we can form a Markov model as shown in Figure 4 and calculate the unavailability of the system using Equation (2). Using the repair time, we can use SAPHIRE to calculate the failure probability as shown in Figure 5. Finally, using the Markov model and SAPHIRE results, we can verify the EMRALD results as shown in Figure 6 and Table 3.

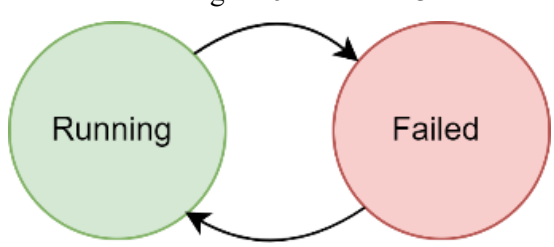


Figure 4. Markov model for single component failure and repair.

$$A(t) = (\mu + \lambda)^{-1} \cdot \{\mu + \lambda \cdot e^{[-(\mu+\lambda)\cdot t]}\} \tag{2}$$

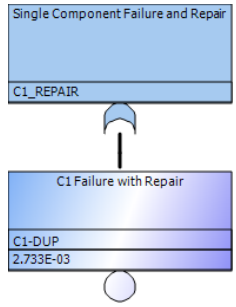


Figure 5. SAPHIRE model for the failure and repair of a single component.

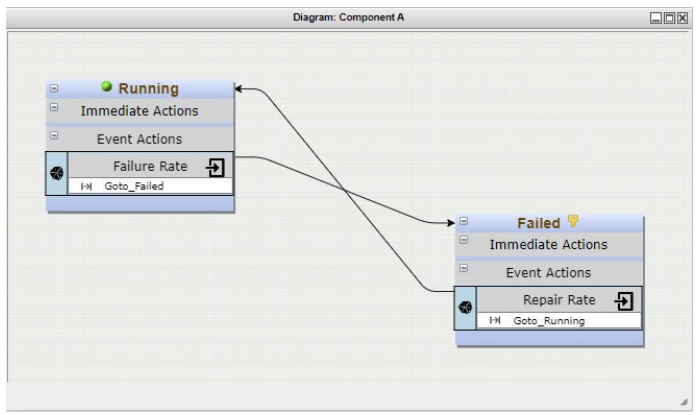


Figure 6. EMRALD model for the failure and repair of a single component.

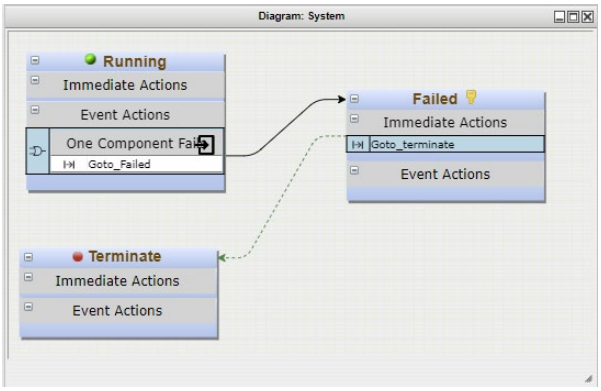
Table 3. Data for validating the failure of a single component.

Test Number	Failure Rate	Repair Rate/ Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ SAPHIRE Failure Probability
1	$\frac{7}{365\ days}$	1 day	1,000 days	100,000	$A(t) = 1.888 \times 10^{-2}$	$A(t) = 1.88172 \times 10^{-2}$
2	$\frac{7}{365\ days}$	1 day	365 days	100,000	$P(f) = 2.84 \times 10^{-3}$	$P(f) = 2.733 \times 10^{-3}$

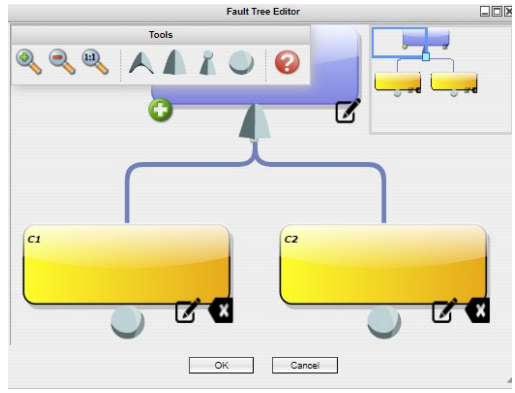
2.2.3 Two Identical Components in Parallel Fail

To test the failure probability and MTTF of two identical components in parallel in EMRALD, as shown in Figure 7, we can use SAPHIRE as shown in Figure 8 and analytical calculations as given in Equation (3). The results are tabulated in Table 4.

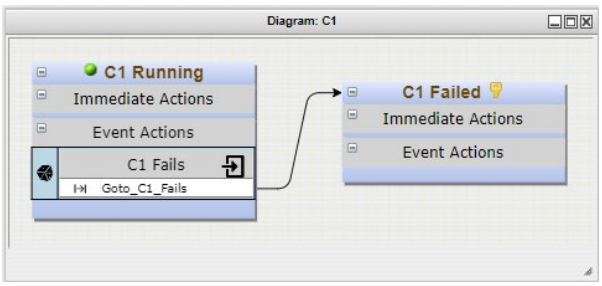
$$MTTF = \frac{2}{3 \cdot \lambda} \tag{3}$$



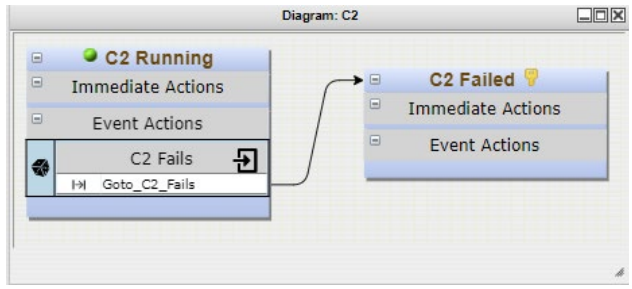
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 7. EMRALD model for the failure of two identical components in parallel.

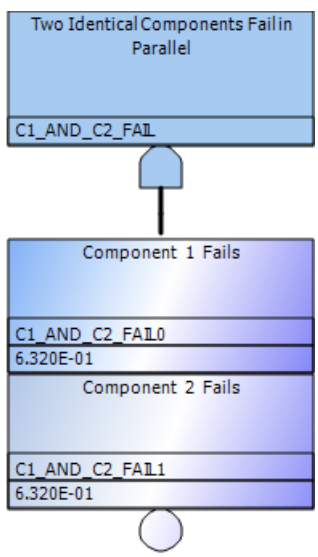


Figure 8. SAPHIRE model for the failure of two identical components in parallel.

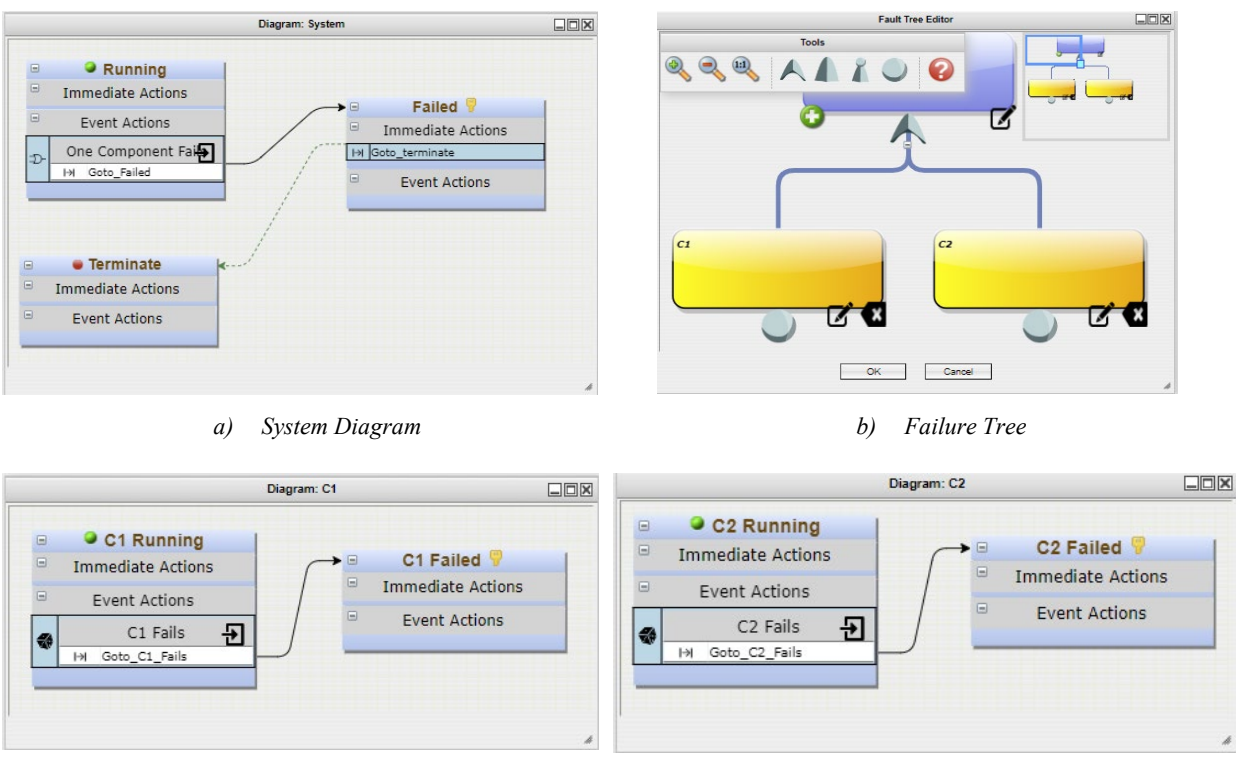
Table 4. Data for validating the failure of two identical components in parallel.

Test Number	Failure Rates of Both Components	Mission Time	Number of Runs	EMRALD Failure Probability	SAPHIRE Failure Probability	EMRALD MTTF	Analytical MTTF
1	$\frac{1}{10\ days}$	100 days	100,000	$P(f) = 9.9995 \times 10^{-1}$	$P(f) = 1 \times 10^{0}$	MTTF = 15 days	MTTF = 15 days
2	$\frac{1}{365\ days}$	365 days	100,000	$P(f) = 3.9728 \times 10^{-1}$	$P(f) = 3.997 \times 10^{-1}$	-	-
3	$\frac{1}{365\ days}$	10,000 days	100,000	$P(f) = 1 \times 10^{0}$	$P(f) = 1 \times 10^{0}$	MTTF = 547 days	MTTF = 547.5 days

The MTTF is not compared in Test 2, because the mission time is not large enough to simulate enough failures to get an accurate result. The mission time was increased in Test 3 and compare the MTTF.

2.2.4 Two Identical Components in Series Fail

To test the failure probability and MTTF of two identical components in series in EMRALD, as shown in Figure 9, we can use SAPHIRE as shown in Figure 10 and analytical calculations as given in Equation (4). The results are tabulated in Table 5.



a) System Diagram *b) Failure Tree*

c) Component 1 *d) Component 2*

Figure 9. EMRALD model for the failure of two identical components in series.

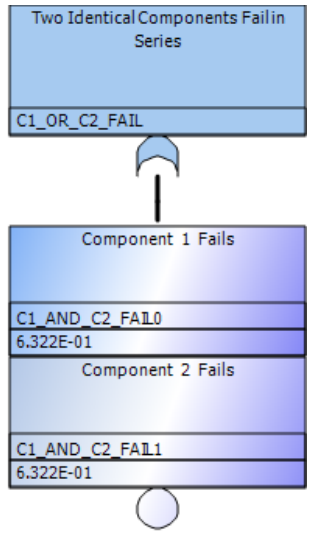


Figure 10. SAPHIRE model for the failure of two identical components in series.

$$MTTF = \frac{1}{2 \cdot \lambda} \tag{4}$$

Table 5. Data for validating the failure of two identical components in series.

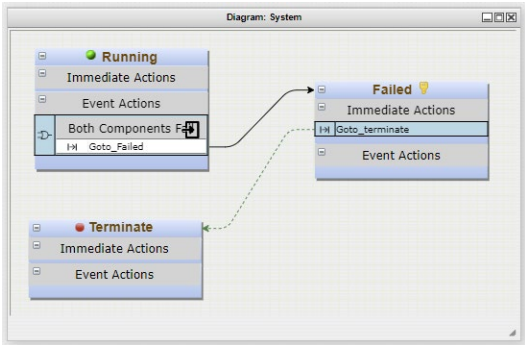
Test Number	Failure Rates of Both Components	Mission Time	Number of Runs	EMRALD Failure Probability	SAPHIRE Failure Probability	EMRALD MTTF	Analytical MTTF
1	$\frac{1}{10\ days}$	100 days	100,000	$P(f) = 1 \times 10^0$	$P(f) = 1 \times 10^0$	MTTF = 5 days	MTTF = 5 days
2	$\frac{1}{365\ days}$	365 days	100,000	$P(f) = 8.638 \times 10^{-1}$	$P(f) = 8.647 \times 10^{-1}$	-	-
3	$\frac{1}{365\ days}$	10,000 days	100,000	$P(f) = 9.9573 \times 10^{-1}$	$P(f) = 1 \times 10^0$	MTTF = 178 days	MTTF = 182.5 days

The MTTF is not compared in Test 2, because the mission time is not large enough to simulate enough failures to get an accurate result. The mission time was increased in Test 3 and compare the MTTF.

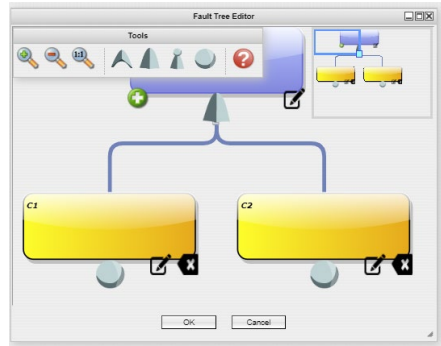
2.2.5 Two Identical Components in Parallel Fail with Common Cause Failure

We can calculate the failure probability of two components in parallel with common cause failure using the equation of the beta factor model given in Equation (5). The beta factor denoted by $\beta, is\ considered$ to be 0.1. Figure 11 shows the EMRALD model for depicting common cause failure, and Table 6 gives the analytical and modeling results.

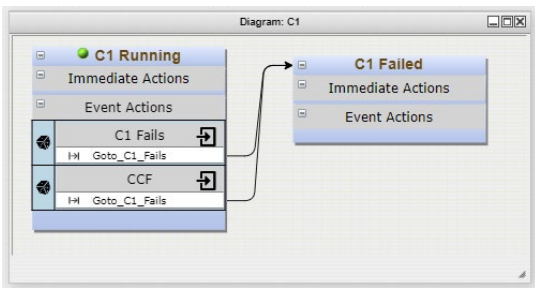
$$P(f) = 2e^{-\lambda t} - e^{-(2-\beta)\lambda t} \tag{5}$$



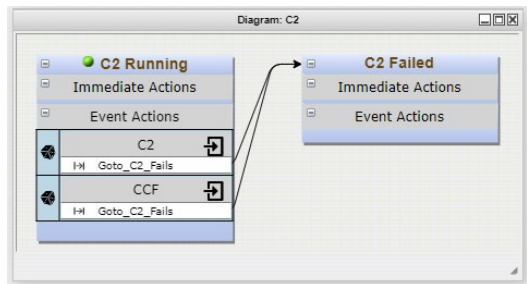
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 11. EMRALD model for the failure and repair of two components in series, one repairperson available.

Table 6. Data for validating the failure and repair of a two components in series, repaired one at a time.

Test Number	Failure Rate	Beta Factor	Mission Time	Number of Runs	EMRALD Failure Probability	Analytical Failure Probability
1	$\frac{1}{100\ days}$	0.1	100 days	100,000	$P(f) = 7.461 \times 10^{-1}$	$P(f) = 5.861 \times 10^{-1}$

2.2.6 Initiating Event with One Engineering Safety Feature

We can verify the results of an event tree model, with one component as the engineering safety feature, in EMRALD. Figure 12 gives the EMRALD model of an event tree that has one component as the engineering safety feature along with an initiating event. Figure 13 shows the SAPHIRE model of the same system, and Equation (6) gives the failure frequency of the core damage state. Table 7 gives the results of the EMRALD and SAPHIRE models.

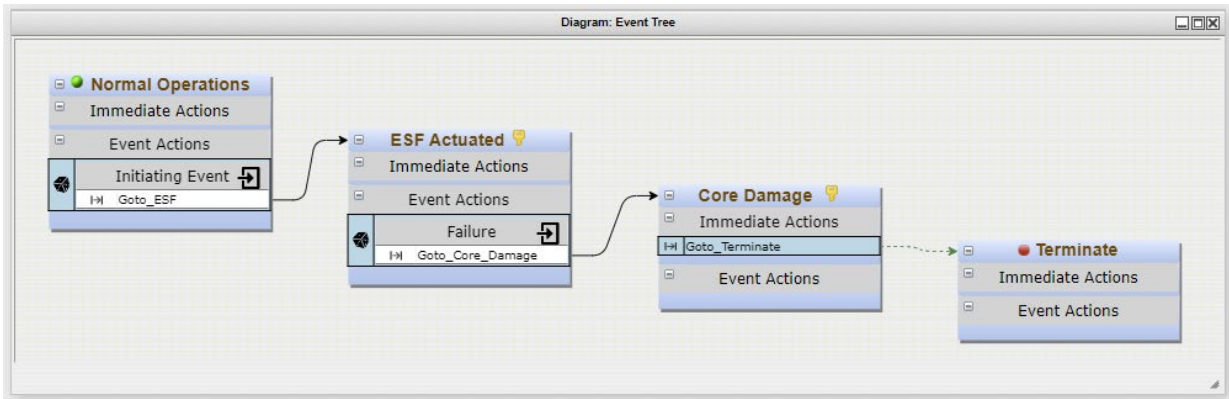


Figure 12. EMRALD model of an event tree with one component as the engineering safety feature.

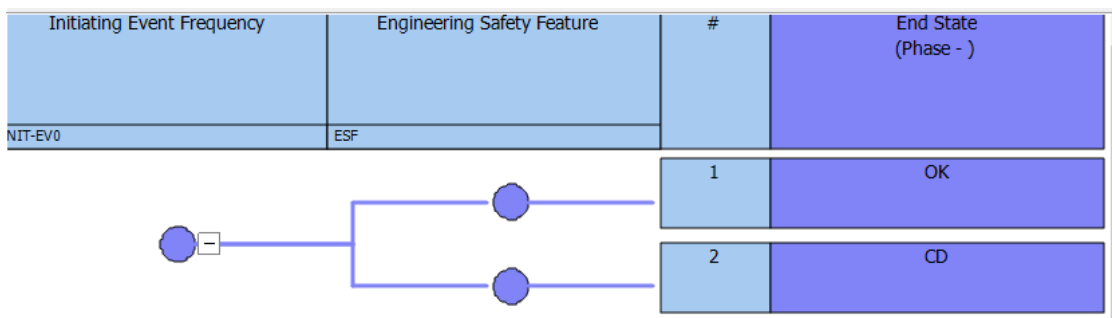


Figure 13. SAPHIRE model of an event tree with one component as the engineering safety feature.

$$\begin{aligned} &Core\ Damage\ Frequency\ (f_{CD}) \\ &\quad = Initiating\ Event\ Frequency(F_0) \\ &\quad * Engineering\ Safety\ Feature\ Failure\ Probability \end{aligned} \tag{6}$$

$$f_{CD} = F_0 \cdot (1 - e^{-\int_0^t \lambda \cdot dt})$$

$$f_{CD} = \frac{0.005}{year} \cdot (1 - e^{-\int_0^{365} \frac{1}{365} \cdot dt})$$

$$f_{CD} = \frac{0.005}{year} \cdot (1 - e^{-\frac{1}{365} \int_0^{365} dt})$$

$$f_{CD} = \frac{0.005}{year} \cdot (1 - e^{-\frac{1}{365}[365-0]})$$

$$f_{CD} = \frac{0.005}{year} \cdot \left(1 - e^{-\frac{1}{365} \cdot 365}\right)$$

$$f_{CD} = \frac{0.005}{year} \cdot 0.6321$$

$$f_{CD} = 0.0031606/year$$

Table 7. Data for validation of failure frequency of an event tree with one component as the engineering safety feature.

Test Number	Initiating Event Frequency	Failure Rate	Mission Time	Number of Runs	EMRALD Failure Frequency	Analytical Failure Frequency
1	$\frac{0.005}{365\ days}$	$\frac{1}{365\ days}$	365 days	10,000,000	$f_{CD} = 3.101 \times 10^{-3}$	$f_{CD} = 3.160 \times 10^{-3}$

2.2.7 Initiating Event with Engineering Safety Feature 1

We can verify the results of an event tree model with an engineering safety feature that has two component sin parallel, in EMRALD. Figure 14 gives the EMRALD model, and Figure 15 shows the SAPHIRE model of the same system, and Equation (7) gives the failure frequency of the core damage state. Table 8 gives the results from the EMRALD and SAPHIRE models.

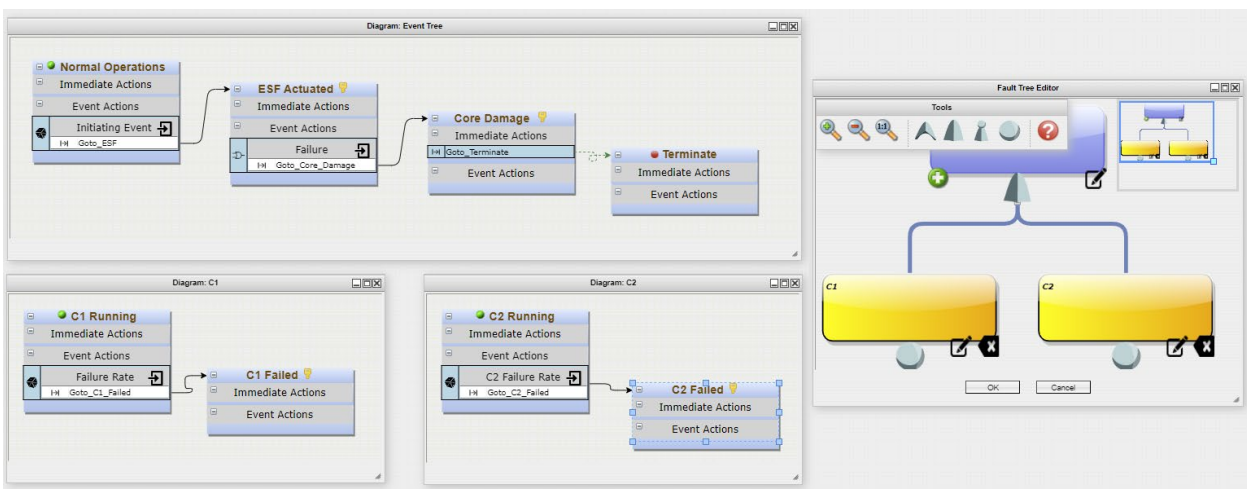


Figure 14. EMRALD model of an event tree with two components in parallel as the engineering safety feature.

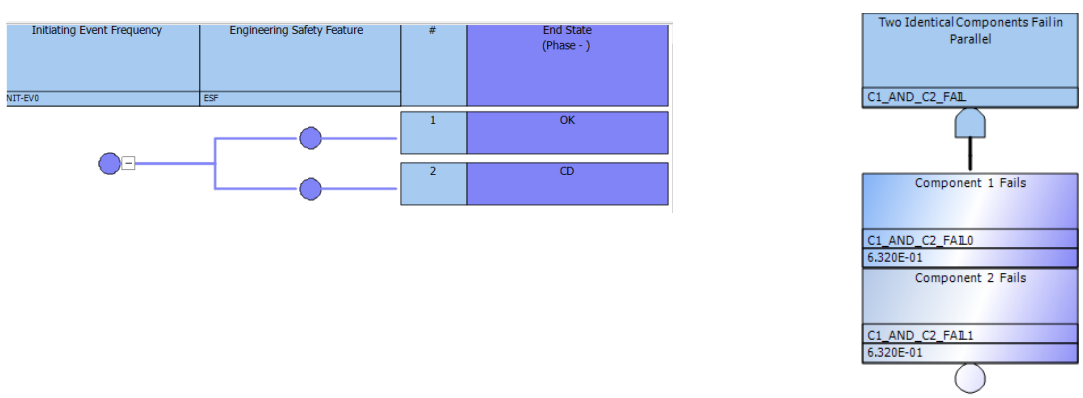


Figure 15. SAPHIRE model of an event tree with two components in parallel as the engineering safety feature.

$$CD = \frac{0.005}{year} \cdot 0.39728 \tag{7}$$

$$CD = 0.0019864/year$$

Table 8. Data for validating the failure frequency of an event tree with two components in parallel as the engineering safety feature.

Test Number	Initiating Event Frequency	Failure Rate	Mission Time	Number of Runs	EMRALD Failure Frequency	Analytical Failure Frequency
1	$\frac{0.005}{365\ days}$	$\frac{1}{365\ days}$	365 days	10,000,000	$f_{CD} = 2.008 \times 10^{-3}$	$f_{CD} = 1.986 \times 10^{-3}$

2.2.8 Initiating Event with Engineering Safety Feature 2

We can verify the results of an event tree model with an engineering safety feature that has two components in series, in EMRALD. Figure 16 gives the EMRALD model and Figure 17 shows the SAPHIRE model of the same system.

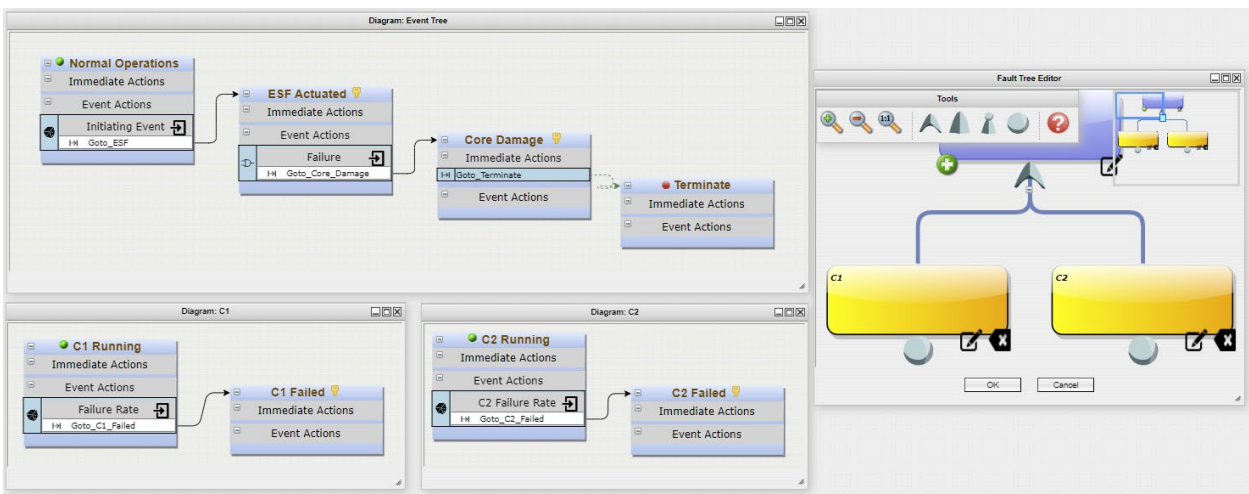


Figure 16. EMRALD model of an event tree with two components in series as the engineering safety feature.

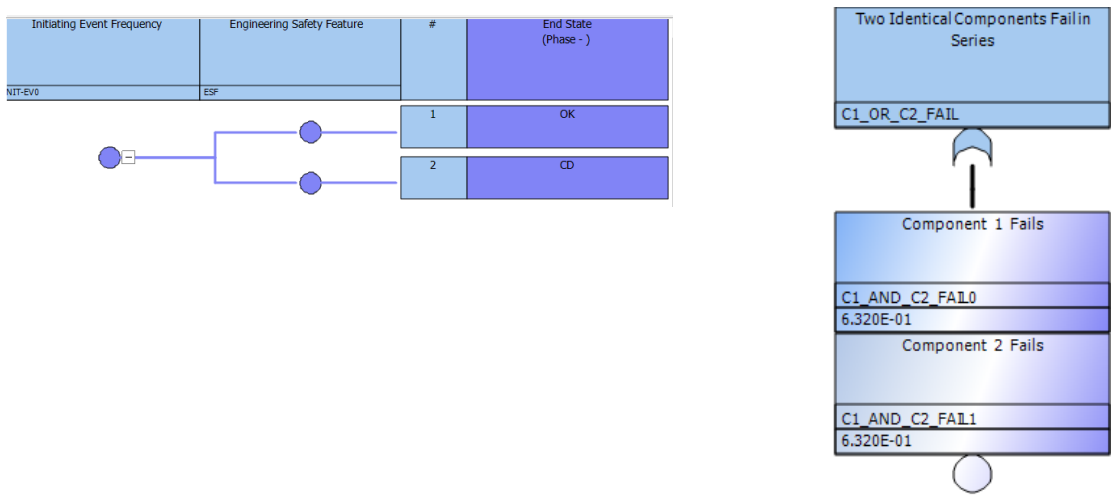


Figure 17. SAPHIRE model of an event tree with two components in series as the engineering safety feature.

Table 9 gives the results from the EMRALD and SAPHIRE models.

Table 9. Data for validating the failure frequency of an event tree with two components in series as the engineering safety feature.

Test Number	Initiating Event Frequency	Failure Rate	Mission Time	Number of Runs	EMRALD Failure Frequency	Analytical Failure Frequency
1	$\frac{0.005}{365\ days}$	$\frac{1}{365\ days}$	365 days	10,000,000	$f_{CD} = 4.51 \times 10^{-3}$	$f_{CD} = 4.32 \times 10^{-3}$

2.3 Dynamic Model Numerical Evaluation Cases

Many simple dynamic cases can be evaluated numerically to verify the EMRALD results. This section evaluates several of those cases. Many of these test cases are compared against the unavailability time calculation. The standard output of EMRALD is the probability of a key state, typically a failure or success state, and time statistics to failure for success. To compare the availability time, a cumulative variable is used to determine the time in the OK states or the inverse. This is summed and averaged for the scenario runs. The results of these validation test cases are summarized in Table 10.

Table 10. Summary of dynamic vs. static evaluation cases.

Mode #	Model Name	Test #	Availability Results	SAPHIRE Results	EMRALD Results
2.3.1	Two Identical Components in Active Parallel Fail and Get Repaired: One Repairperson Available	1	A(t) = 1×10^{-3}		A(t) = 8.264×10^{-3}
		2	A(t) = 1.6×10^{-2}		A(t) = 1.6×10^{-2}
2.3.2	Two Identical Components in Active Parallel Fail and Get Repaired: Two Repairpersons Available	1	A(t) = 8.235×10^{-3}		A(t) = 8.2644628×10^{-3}
		2	A(t) = 8.275×10^{-3}		A(t) = 8.2644628×10^{-3}
		3		P(f) = 8.263×10^{-3}	P(f) = 8.025×10^{-3}
2.3.3	Two Identical Components, One Active and One in	1	A(t) = 5.5×10^{-2}		A(t) = 9.009×10^{-3}

Mode #	Model Name	Test #	Availability Results	SAPHIRE Results	EMRALD Results
	Standby Fail and Get Repaired, One Repairperson Available				
		2	A(t) = 8.985×10^{-3}		A(t) = 9.009×10^{-3}
2.3.4	Two Identical Components, One Active and One in Standby Fail and Get Repaired, Two Repairpersons Available	1	A(t) = 4.541×10^{-3}		A(t) = 4.5248×10^{-3}
		2	A(t) = 4.524×10^{-3}		A(t) = 4.5248×10^{-3}
2.3.5	Two Identical Components in Series Fail and Get Repaired: One Repairperson Available	1	A(t) = 0.0123×10^{-1}		A(t) = 1.8033×10^{-1}
		2	A(t) = 1.8001×10^{-1}		A(t) = 1.8033×10^{-1}
2.3.6	Two Identical Components in Series Fail and Get Repaired: Two Repairpersons Available	1	A(t) = 1.624×10^{-1}		A(t) = 1.735×10^{-1}
		2	A(t) = 1.735×10^{-1}		A(t) = 1.735×10^{-1}
		3		P(f) = 1.622×10^{-1}	P(f) = 1.711×10^{-1}

Model #	Model Name	Test #	SAPHIRE Results	EMRALD Results
2.3.1	Two Identical Components in Active Parallel Fail and Get Repaired, One Repairperson Available	1	A(t) = 1×10^{-3}	A(t) = 8.264×10^{-3}
		2	A(t) = 1.6×10^{-2}	A(t) = 1.6×10^{-2}

Mode #	Model Name	Test #	Availability Results	SAPHIRE Results	EMRALD Results
2.3.2	Two Identical Components in Active Parallel Fail and Get Repaired, Two Repairpersons Available		1	A(t) = 8.235×10^{-3}	A(t) = 8.2644628×10^{-3}
			2	A(t) = 8.275×10^{-3}	A(t) = 8.2644628×10^{-3}
			3	P(f) = 8.263×10^{-3}	P(f) = 8.025×10^{-3}
2.3.3	Two Identical Components, One Active and One in Standby Fail and Get Repaired, One Repairperson Available		1	A(t) = 5.5×10^{-2}	A(t) = 9.009×10^{-3}
			2	A(t) = 8.985×10^{-3}	A(t) = 9.009×10^{-3}
2.3.4	Two Identical Components, One Active and One in Standby Fail and Get Repaired, Two Repairpersons Available		1	A(t) = 4.541×10^{-3}	A(t) = 4.5248×10^{-3}
			2	A(t) = 4.524×10^{-3}	A(t) = 4.5248×10^{-3}
2.3.5	Two Identical Components in Series Fail and Get Repaired, One Repairperson Available		1	A(t) = 0.0123×10^{-1}	A(t) = 1.8033×10^{-1}
			2	A(t) = 1.8001×10^{-1}	A(t) = 1.8033×10^{-1}
2.3.6	Two Identical Components in Series Fail and Get Repaired, Two Repairpersons Available		1	A(t) = 1.624×10^{-1}	A(t) = 1.735×10^{-1}
			2	A(t) = 1.735×10^{-1}	A(t) = 1.735×10^{-1}
			3	P(f) = 1.622×10^{-1}	P(f) = 1.711×10^{-1}

2.3.1 Two Identical Components in Active Parallel Fail and Get Repaired, One Repairperson Available

We can calculate the availability of two components in parallel that fail and can be repaired one at a time using a Markov model as shown in Figure 18, whose availability equation is given in Equation (8) [22]. The EMRALD model for the system is given in Figure 19. However, its results are significantly higher when compared to the analytical calculations as shown in Table 11, Test 1. Therefore, another EMRALD model was developed which evaluates a more standard Markov model of the system rather

than decomposing the system into separate components, as shown in Figure 20. The results of the second EMRALD model are close to the analytical unavailability as shown in Table 12, Test 2. Further investigation needs to be done as to why the first model used was not equivalent.

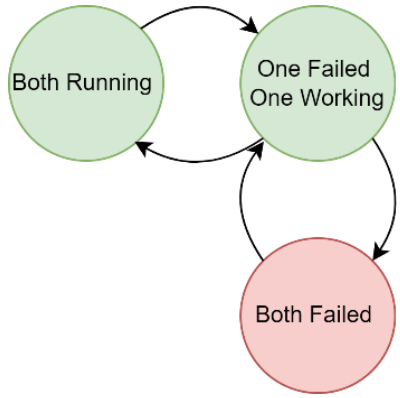
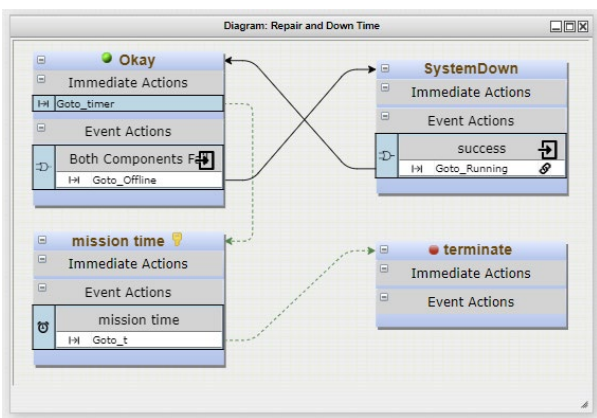


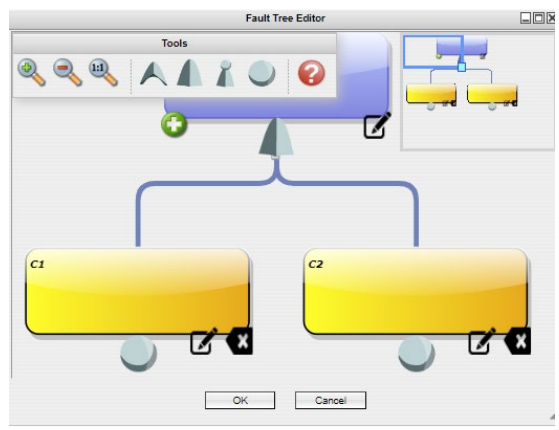
Figure 18. Markov model for failure of two components in parallel, one repairperson available.

$$A(t) = \frac{\mu^2 + 2\mu\lambda}{\mu^2 + 2\lambda\mu + 2\lambda^2} - \frac{2\lambda^2\{s_2 e^{(s_1 t)} - s_1 e^{(s_2 t)}\}}{s_1 s_2 (s_1 - s_2)}; \tag{8}$$

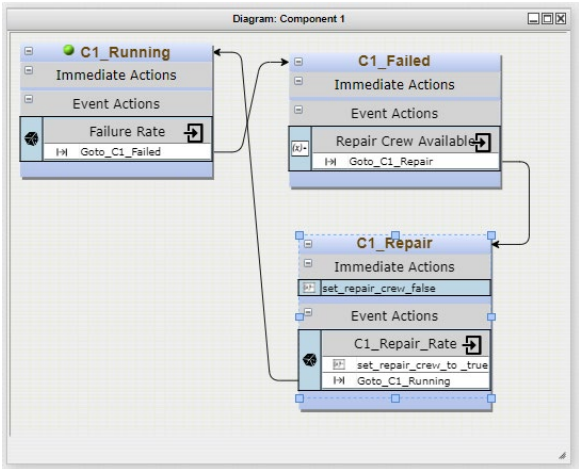
$$s_1 = -0.5\left(2\mu + 3\lambda + \sqrt{4\lambda\mu + \lambda^2}\right);\ s_2 = -0.5\left(2\mu + 3\lambda - \sqrt{4\lambda\mu + \lambda^2}\right)$$



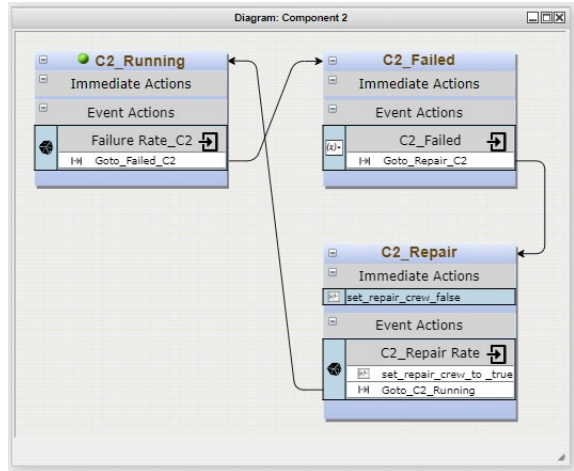
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 19. EMRALD model for the failure and repair of two components in parallel, one repairperson available.

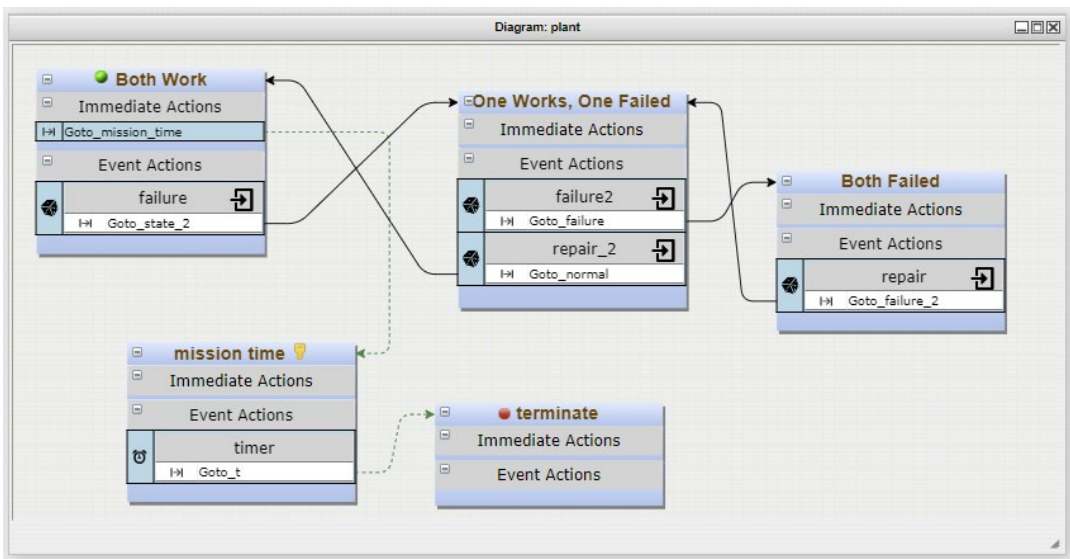


Figure 20. EMRALD model of the Markov model for the failure and repair of two components in parallel, one repairperson available.

Table 11. Data for validating the failure and repair of two components in parallel, repaired one at a time.

Test Number	Failure Rate	Repair Rate/Time	Mission Time	Number of Runs	EMRALD Unavailability	Analytical Unavailability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 1 \times 10^{-3}$	$A(t) = 8.264 \times 10^{-3}$
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 1.6 \times 10^{-2}$	$A(t) = 0.8264 \times 10^{-2}$

2.3.2 Two Identical Components in Active Parallel Fail and Get Repaired, Two Repairpersons Available

The availability of two components in parallel that fail and can be repaired simultaneously can be calculated using the failure and repair rate and a Markov model, as shown in Figure 21. The availability equation is given in Equation (9). The EMRALD model for the system is given in Figure 22, and another EMRALD model, which evaluates a more standard Markov model of the system, rather than the decomposition of the system into separate components, is shown in Figure 23. The results of the second EMRALD model are close to the analytical unavailability as shown in Table 12, Test 2, versus the results of the first EMRALD model given in Table 12, Test 1. We can also compare the results of a SAPHIRE model and the corresponding EMRALD model when the failure rates and repair duration are considered. The SAPHIRE model is shown in Figure 24, and the EMRALD model is the same as Figure 21 other than the repair events being of the timer type instead of a rate.

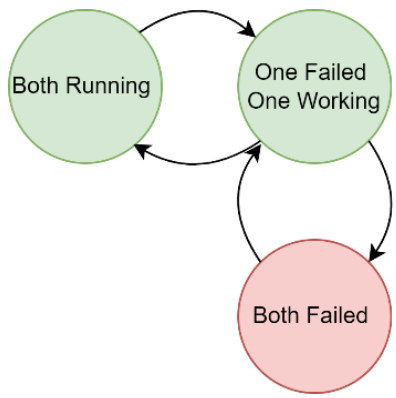
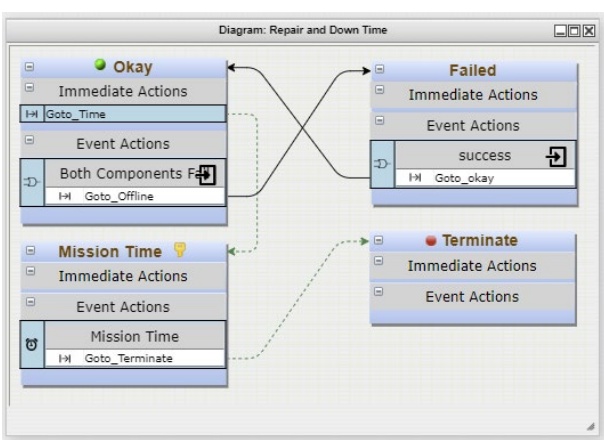


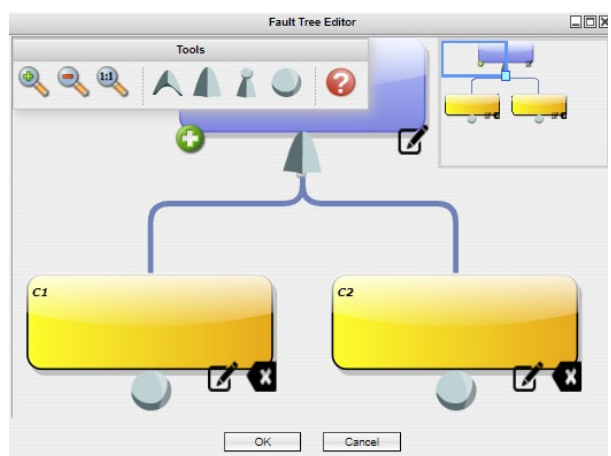
Figure 21. Markov model for failure of two components in parallel, two repairpersons available.

$$A(t) = \frac{\mu^2 + 2\mu\lambda}{\mu^2 + 2\lambda\mu + \lambda^2} - \frac{2\lambda^2\{s_2 e^{(s_1 t)} - s_1 e^{(s_2 t)}\}}{s_1 s_2 (s_1 - s_2)}; \tag{9}$$

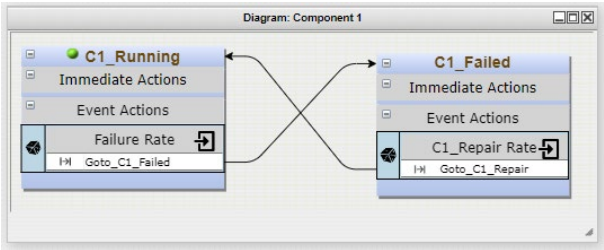
$$s_1 = -2(\mu + \lambda);\ s_2 = -(\mu + \lambda)$$



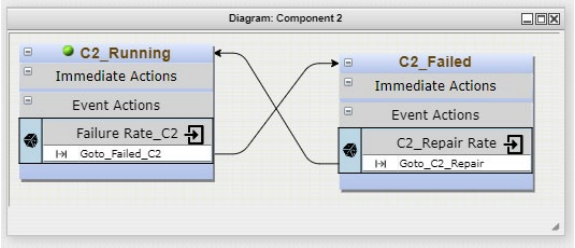
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 22. EMRALD model for the failure and repair of two parallel components, two repairpersons available.

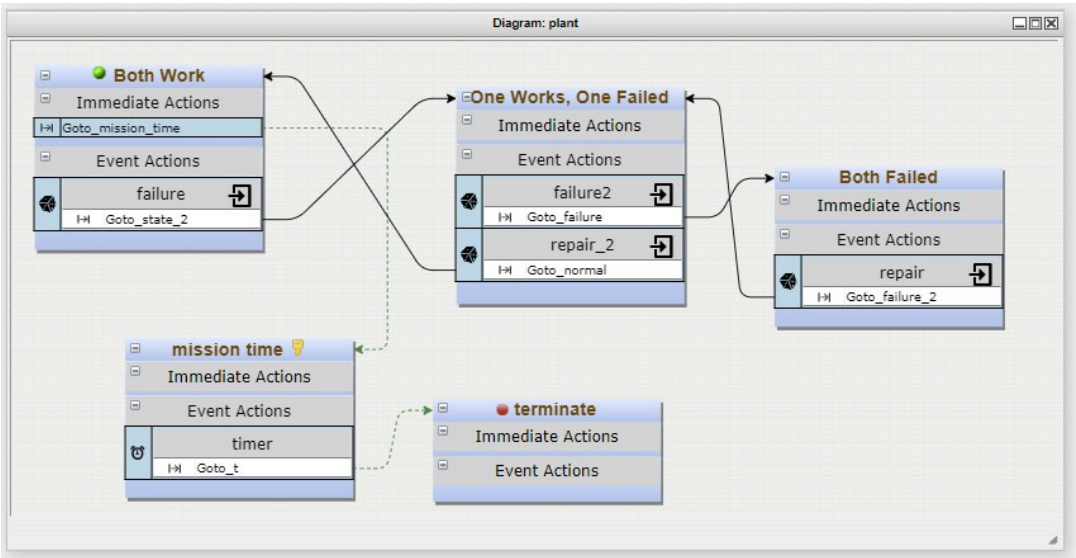


Figure 23. EMRALD model of the Markov model for the failure and repair of two parallel components, two repairpersons available.

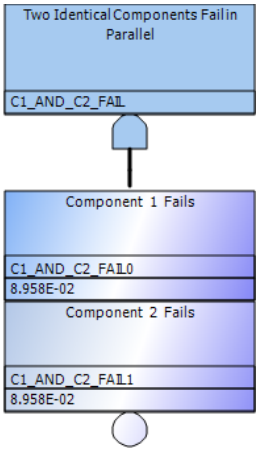


Figure 24. SAPHIRE model for failure of two identical components in parallel, which can be repaired simultaneously.

Table 12. Data for validating the failure and repair of two components in parallel, repaired simultaneously.

Test Number	Failure Rate	Repair Rate/ Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ SAPHIRE Failure Probability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	A(t) = 8.235× 10^{-3}	A(t) = 8.2644628 × 10^{-3}
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	A(t) = 8.275 × 10^{-3}	A(t) = 8.2644628 × 10^{-3}
3	$\frac{1}{10\ days}$	1 *day*	1,000 days	100,000	P(f) = 8.263 × 10^{-3}	P(f) = 8.025 × 10^{-3}

2.3.3 Two Identical Components, One Active and One in Standby, Fail and Get Repaired, One Repairperson Available

The availability of two components, one active and one in standby, that fail and can be repaired one at a time can be calculated using a Markov model as shown in Figure 25, whose availability equation is given in Equation (10) [22]. The EMRALD model for the system is given in Figure 29, and another EMRALD model, which evaluates a more standard Markov model of the system rather than the decomposition of the system into separate components, is shown in Figure 27. Support diagrams for components are shown in Figure 26. The results of the second EMRALD model are close to the analytical unavailability as shown in Table 13, Test 2, versus the results of the first EMRALD model given in Table 13, Test 1. Further investigation needs to be done as to why the modeling method first used was not equivalent.

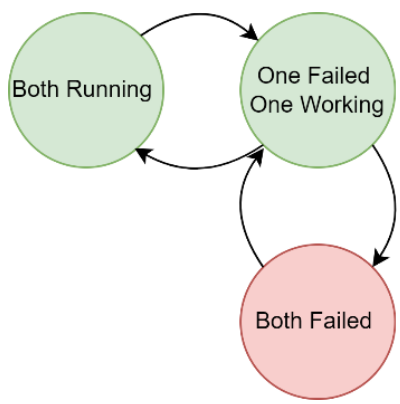
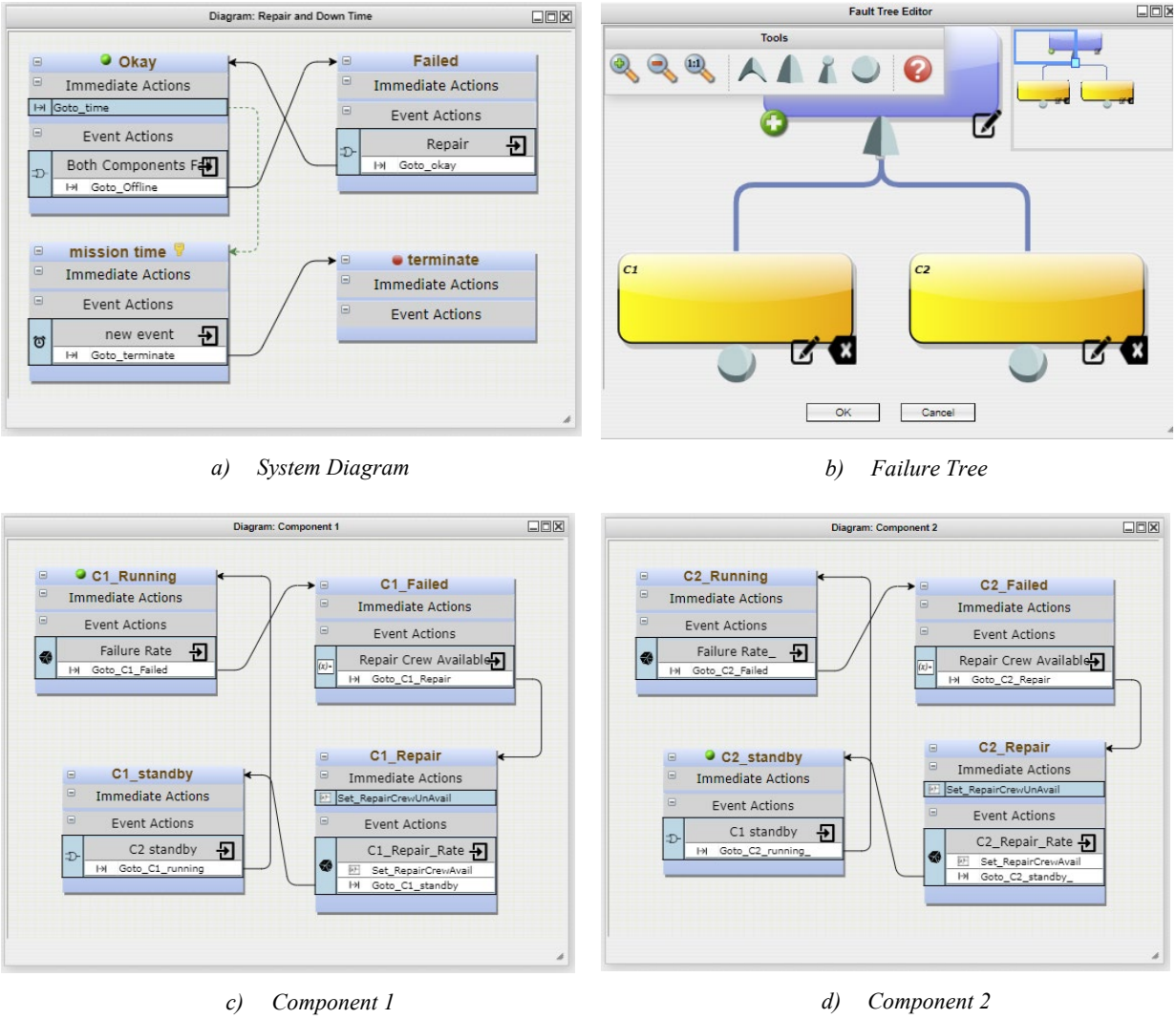


Figure 25. Markov model for failure of two components, one active and one in standby, one repairperson available.

$$A(t) = \frac{\mu^2 + \mu\lambda}{\mu^2 + \lambda\mu + \lambda^2} - \frac{\lambda^2\{s_2 e^{(s_1 t)} - s_1 e^{(s_2 t)}\}}{s_1 s_2 (s_1 - s_2)}; \tag{10}$$

$$s_1 = -(\mu + \lambda + \sqrt{2\lambda\mu});\ s_2 = -(\mu + \lambda - \sqrt{2\lambda\mu})$$



a) System Diagram *b) Failure Tree*

c) Component 1 *d) Component 2*

Figure 26. EMRALD model for the failure and repair of one active and one standby component, one repairperson available.

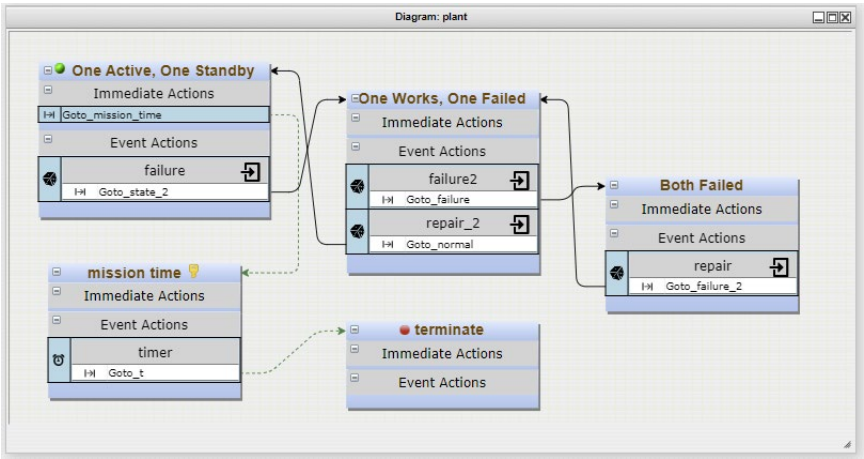


Figure 27. EMRALD model of Markov model for the failure and repair of one active and one standby component, one repairperson available.

Table 13. Data for validating the failure and repair of two components, one active and one in standby, repaired one at a time.

Test Number	Failure Rate	Repair Rate/ Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ Failure Probability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	A(t) = 5.5×10^{-2}	A(t) = 9.009×10^{-3}
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	A(t) = 8.985×10^{-3}	A(t) = 9.009×10^{-3}

2.3.4 Two Identical Components, One Active and One in Standby, Fail and Get Repaired, Two Repairpersons Available

The availability of two components, one active and one in standby, that fail and can be repaired simultaneously can be calculated using a Markov model as shown in Figure 28, whose availability equation is given in Equation (11) [22]. The EMRALD models for the system are given in Figure 29 and Figure 30, where the first model includes all the components and their states while the second model is the Markov model representation of the system. Table 14 gives a comparison of the analytical and EMRALD results.

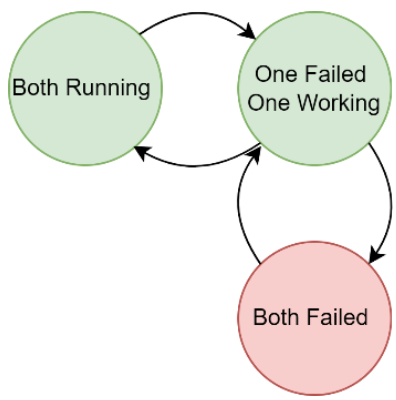
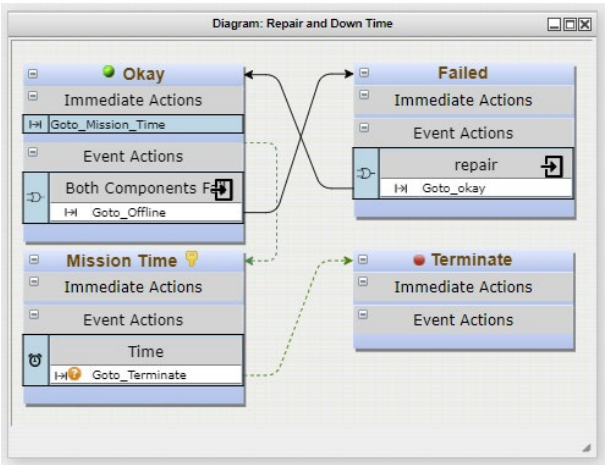


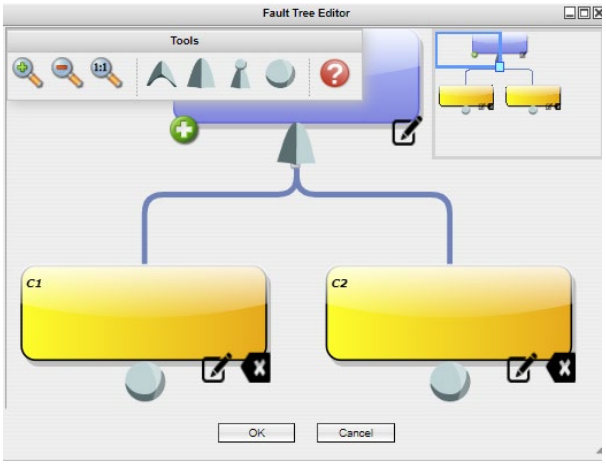
Figure 28. Markov model for failure of two components, one active and one in standby, two repairpersons available.

$$A(t) = \frac{2\mu^2 + 2\mu\lambda}{2\mu^2 + 2\lambda\mu + \lambda^2} - \frac{\lambda^2\{s_2 e^{(s_1 t)} - s_1 e^{(s_2 t)}\}}{s_1 s_2 (s_1 - s_2)}; \tag{11}$$

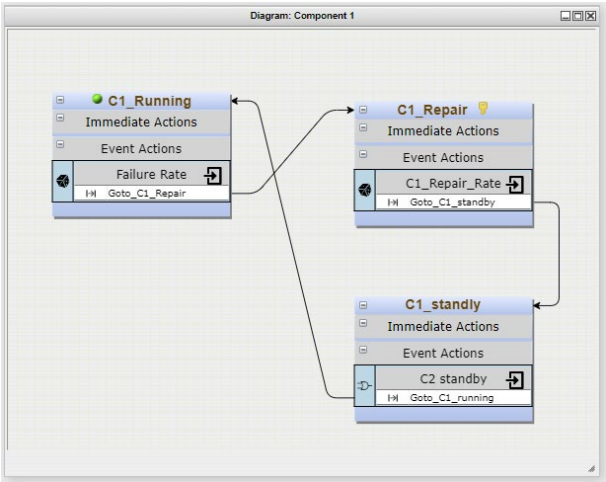
$$s_1 = -0.5\left(3\mu + 2\lambda + \sqrt{4\lambda\mu + \mu^2}\right);\ s_2 = -0.5\left(3\mu + 2\lambda - \sqrt{4\lambda\mu + \mu^2}\right)$$



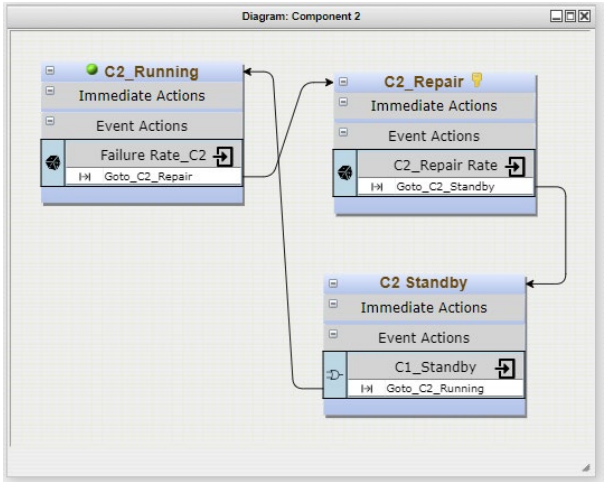
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 29. EMRALD model for the failure and repair of one active and one standby component, two repairpersons available.

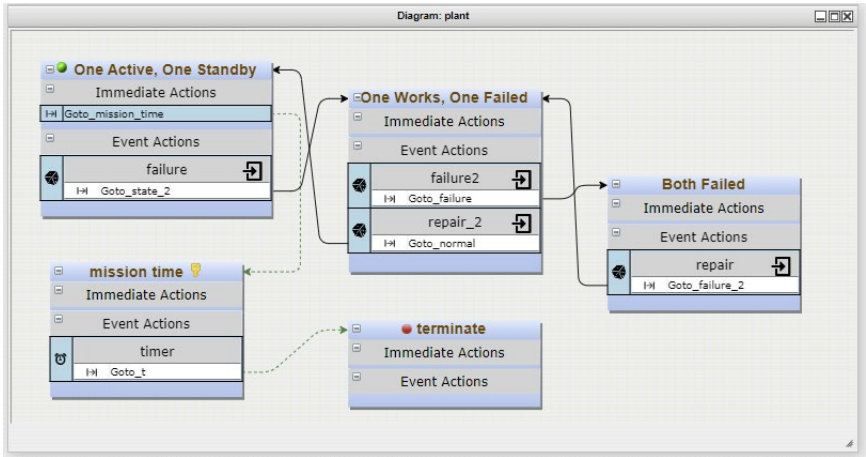


Figure 30. EMRALD model of Markov model for the failure and repair of one active and one standby component, two repairpersons available.

Table 14. Data for validating the failure and repair of two components, one active and one in standby, repaired simultaneously.

Test Number	Failure Rate	Repair Rate/ Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ Failure Probability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 3.911 \times 10^{-3}$	$A(t) = 4.5248 \times 10^{-3}$
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 4.524 \times 10^{-3}$	$A(t) = 4.5248 \times 10^{-3}$

2.3.5 Two Identical Components in Series Fail and Get Repaired, One Repairperson Available

The availability of two components in series that fail and can be repaired one at a time can be calculated using a Markov model as shown in Figure 31, whose availability equation is given in Equation (12) [22]. The EMRALD models for the system are given in Figure 32 and Figure 33, where the first model includes all the components and their states while the second model is the Markov model representation of the system. Table 15 gives a comparison of the analytical and EMRALD results. The results of the second EMRALD model are closer to the analytical unavailability as shown in Table 15, Test 2, versus the results of the first EMRALD model given in Table 15, Test 1, which compared quite low to the analytical unavailability. However, further investigation needs to be done as to why both modeling methods used were not equivalent.

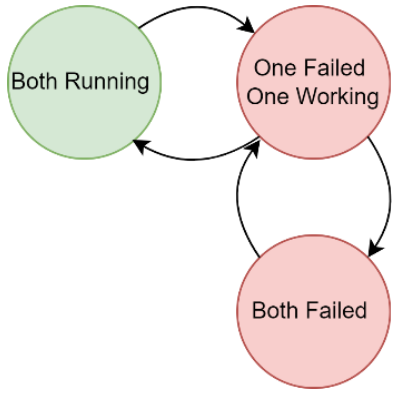
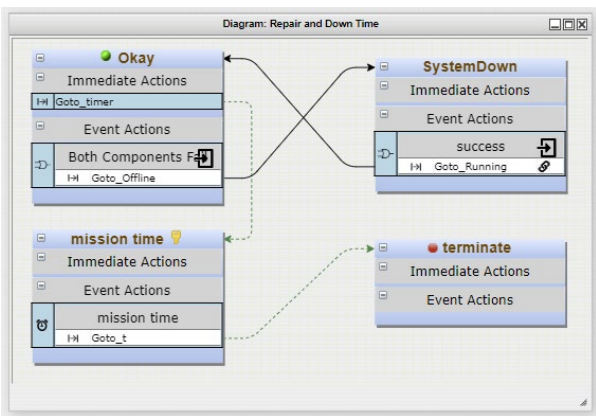
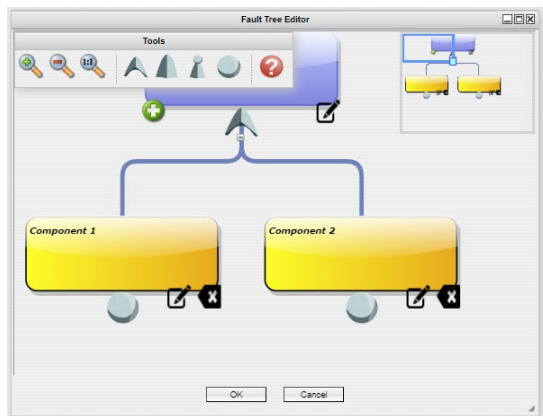


Figure 31. Markov model for the failure and repair of two components in series, one repairperson available.

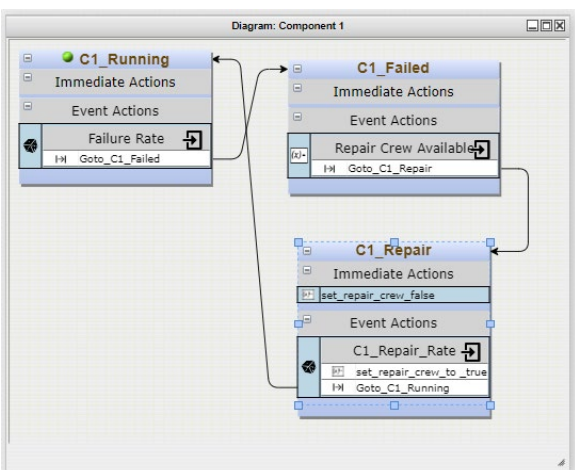
$$A(t) = \frac{e^{s_1 t}(s_1(2\mu + \lambda) + s_1^2 + \mu^2)}{s_1(s_1 - s_2)} + \frac{e^{s_2 t}(s_2(2\mu + \lambda) + s_2^2 + \mu^2)}{s_1(s_1 - s_2)} + \frac{\mu^2}{s_1 s_1} \tag{12}$$



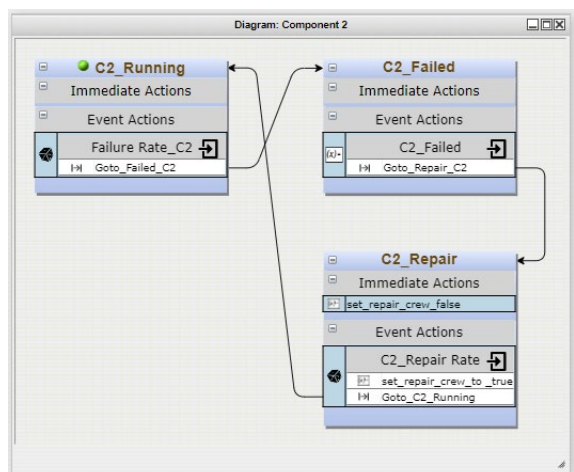
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 32. EMRALD model for the failure and repair of two components in series, one repairperson available.

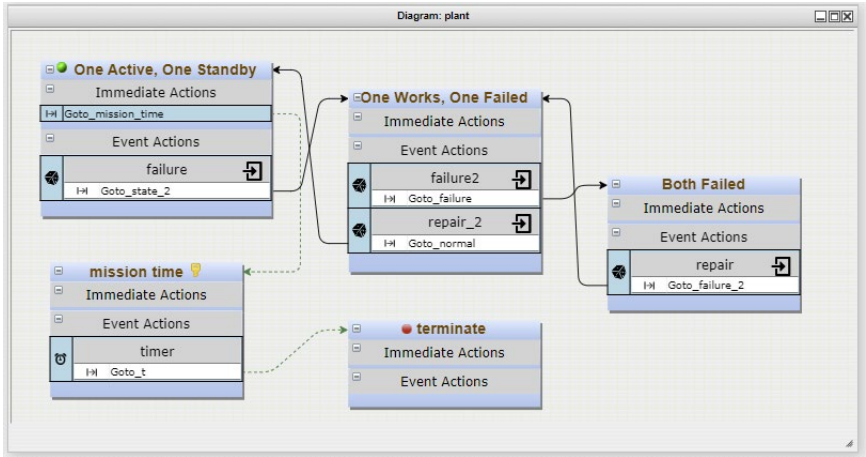


Figure 33. EMRALD model of Markov model for the failure and repair of two components in series, one repairperson available.

Table 15. Data for validating the failure and repair of two components in series, repaired one at a time.

Test Number	Failure Rate	Repair Rate/ Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ Failure Probability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1000 days	100000	$A(t) = 0.0123 \times 10^{-1}$	$A(t) = 1.8033 \times 10^{-1}$
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1000 days	100000	$A(t) = 1.8001 \times 10^{-1}$	$A(t) = 1.8033 \times 10^{-1}$

2.3.6 Two Identical Components in Series Fail and Get Repaired, Two Repairpersons Available

The availability of two components in series that fail and can be repaired one at a time can be calculated using a Markov model as shown in Figure 34, whose availability equation is given in Equation (13) [22]. The EMRALD models for the system are given in Figure 35 and Figure 36, where the first model includes all the components and their states while the second model is the Markov model representation of the system. Table 16 gives a comparison of the analytical and EMRALD results. The results of the second EMRALD model are close to the analytical unavailability as shown in Table 16, Test 2, versus the results of the first EMRALD model given in Table 16, Test 1, which compared low to the analytical unavailability. Further investigation needs to be done as to why the modeling method first used was not equivalent. The components, with a failure rate and repair time, can be modeled in SAPHIRE as shown in Figure 37. The results are compared to the corresponding EMRALD model as shown in Table 16, Test 3.

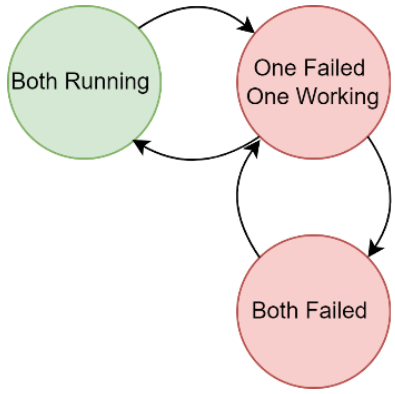
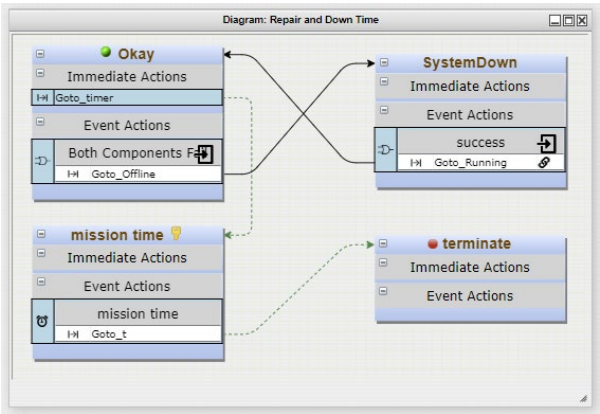
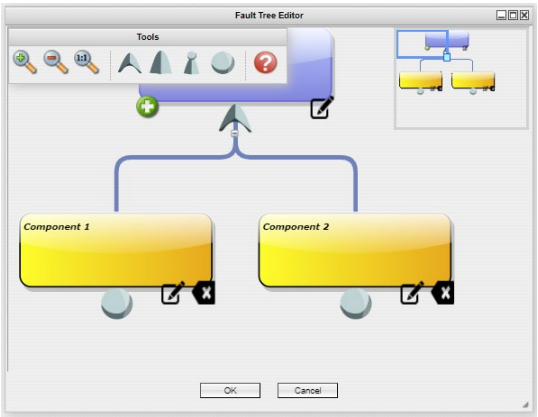


Figure 34. Markov model for the failure and repair of two components in series, one repairperson available.

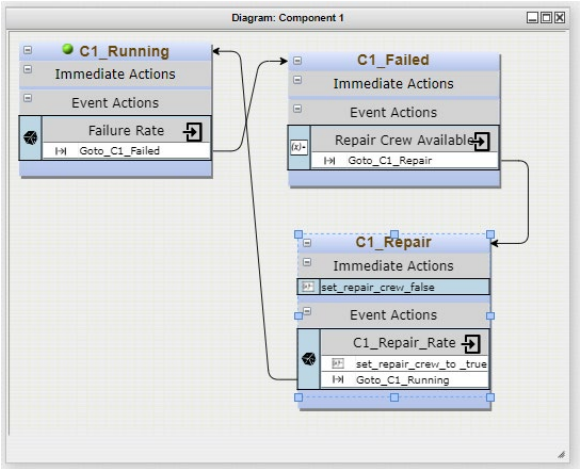
$$A(t) = \frac{e^{s_1 t}(s_1(3\mu + \lambda) + s_1^2 + 2\mu^2)}{s_1(s_1 - s_2)} + \frac{e^{s_2 t}(s_2(3\mu + \lambda) + s_2^2 + 2\mu^2)}{s_1(s_1 - s_2)} + \frac{2\mu^2}{s_1 s_1} \tag{13}$$



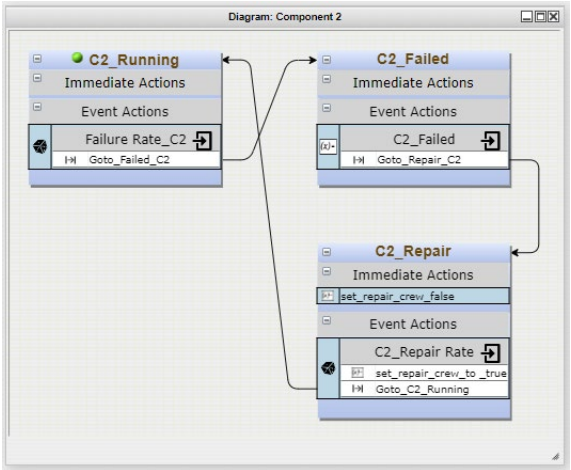
a) System Diagram



b) Failure Tree



c) Component 1



d) Component 2

Figure 35. EMRALD model for the failure and repair of two components in series, one repairperson available.

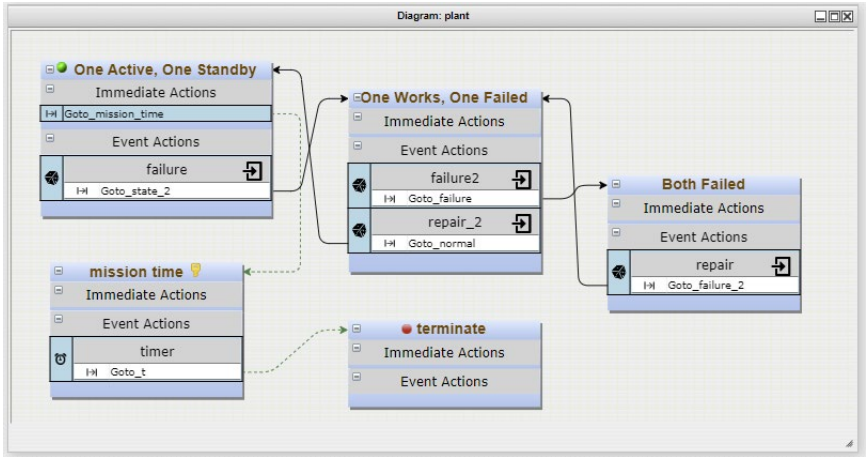


Figure 36. EMRALD model of Markov model for the failure and repair of two components in series, one repairperson available.

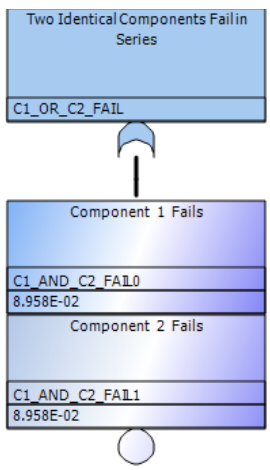


Figure 37. SAPHIRE model for the failure of two identical components in series, which can be repaired one a time.

Table 16. Data for validating the failure and repair of two components in series, repaired one at a time.

Test Number	Failure Rate	Repair Rate/Time	Mission Time	Number of Runs	EMRALD Unavailability/ Failure Probability	Analytical Unavailability/ Failure Probability
1	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 1.624 \times 10^{-1}$	$A(t) = 1.735 \times 10^{-1}$
2	$\frac{1}{10\ days}$	$\frac{1}{day}$	1,000 days	100,000	$A(t) = 1.735 \times 10^{-1}$	$A(t) = 1.735 \times 10^{-1}$
3	$\frac{1}{10\ days}$	$1\ day$	1,000 days	100,000	$P(f) = 1.622 \times 10^{-1}$	$P(f) = 1.711 \times 10^{-1}$

3. EMRALD ENHANCEMENTS

Several modifications to improve the capabilities and usability of EMRALD were added. Many other features not listed here have been added and funded under specific projects; more requests would have also been added, but development resources were limited due to other project needs.

3.1 Solve Engine

- Debug Log – Several items were added to the debug log to make it easier to understand and follow the execution of a model and find any model issues.
- Recent Models – The solver user interface now remembers recent models that were loaded and the settings used for the most recent solving of that model. This greatly reduces the time for loading and running a model.
- Sankey Result Option – A button and code were added to allow the user to directly open the Sankey time results previously run in a browser vs. going back to the web graphical user interface (GUI).
- Variable Options – The option to use variables in almost all parameter fields of a model was added.
- New Distribution Types – Several other distribution types were added including Weibull, Triangular, Gamma, and Gompertz.
- Bug Fixes – Various bug fixes found during the validation work.

3.2 Modeling User Interface

- Templates – The ability to define and use templates was added. The user can create a template from an existing diagram and then create a new diagram from it. Any conflicts cause a dialogue to appear, helping the user to resolve the conflict. This tool was built upon for another project to add additional template options for such categories and saving. These changes became part of the source code.
- Copy/Paste – Added the ability to copy and paste diagrams between different EMRALD models.
- Save and Naming – Added the ability to synchronize the model's name when loading.
- Success or Failure Tree Options – EMRALD now provides the user several different options in evaluating a logic tree. They can treat the same logic tree as a success tree or failure tree and choose whether to trigger the event on a true or false top value.
- Results Visualization – This is discussed in detail in Section 4.

3.3 Future Updates

The biggest update needed for EMRALD is the web-based framework. The EMRALD web user interface (UI) was written in AngularJS which is no longer supported. This does not cause security issues as it is not connected to any authentication or databases. However, it could soon cause usage issues. Plans for upgrading to a supported platform are underway and are slated to begin in FY 2024.

Other features desired by users include uncertainty quantification methods, importance measures, and initiating event frequency options.

4. EMRALD RESULTS AND VISUALIZATION

Dynamic PRA is defined as a PRA with time-dependent effects by integrating them directly into the computer model [23]. In a static PRA, the order of events in an event tree is determined by the analyst a priori. In contrast, dynamic PRA can integrate physical models into the risk assessment to represent a dynamic system. It facilitates the inclusion of dynamic events such as human interactions, digital control systems, passive components, etc. [24]. By using physical models to capture the impact of the aging

process, dynamic PRA can be used to account for the impact of the aging of structures. We can use dynamic PRA to get time to recovery and time required to reach an undesired end state [25].

4.1 Standard Result Capabilities

An example of the EMRALD UI as it is running and displaying the results is shown in Figure 38. The standard results in EMRALD output the number of times the simulations ended in one of the model's key states. The probability is determined by dividing this number by the number of simulations runs. This is similar to the probability calculated by classical PRA methods except it is the encountered simulation conditions and not a numerical calculation. EMRALD also provides failed components that could have contributed to the key end state. These are similar to classical PRA cut sets except that they may or may not directly contribute to the result. An additional component may happen to have failed in that simulation run but not help cause the overall outcome.

As the EMRALD simulations run, the UI updates with the current status of key states and variable values. This allows the user to see the progress of the runs. The standard output results are saved to a text file and are similar to that shown in the UI.

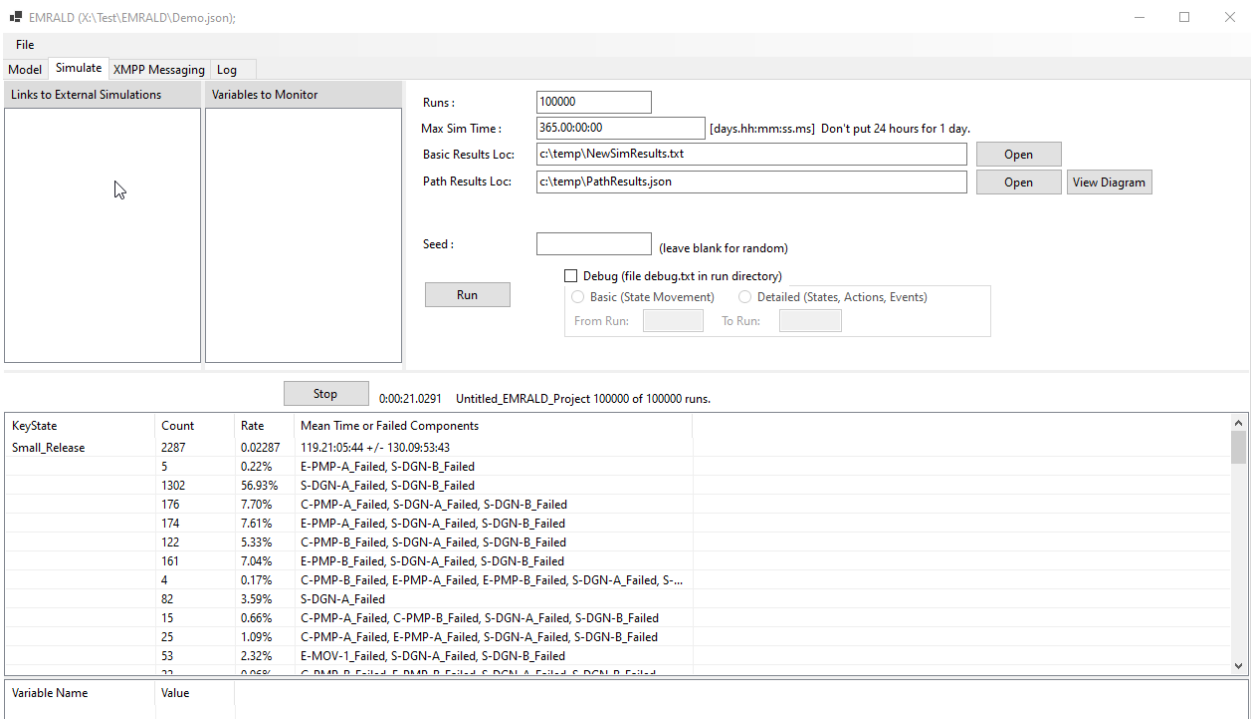


Figure 38. Example of EMRALD UI with results.

4.2 Graphical Base Results

4.2.1 Path Results Data

The path results in EMRALD store the states the modeler is concerned about, called key states. If the model is in any of those key states when a simulation run finishes the path and timing data for that run are saved. As the simulation runs, each state the model goes through, including the events and movement action, the time, and the variable values at that time, are logged for the key states. This is saved in a JavaScript Object Notation (JSON) format file that can easily be parsed, data mined, or looked at by the user. Statistics on the timing of each state, such as mean, min, max, and standard deviation, are also calculated and saved as part of the state path information.

4.2.2 Time-Based Sankey

Sankey diagrams are a type of flow diagram used to visualize branching flows between nodes as shown in Figure 39. The sizes of the links between nodes are scaled to the proportional amount of flow between the source and destination nodes. A Sankey diagram is used to visualize EMRALD results, in which the nodes represent states visited on the way to reaching each key state in the simulation. Transitions between states are represented by a link between those nodes; the size of which is proportional to the number of simulations runs in which that path between states occurred. Hovering over the different states shows the statistics for that state, as shown in Figure 39.

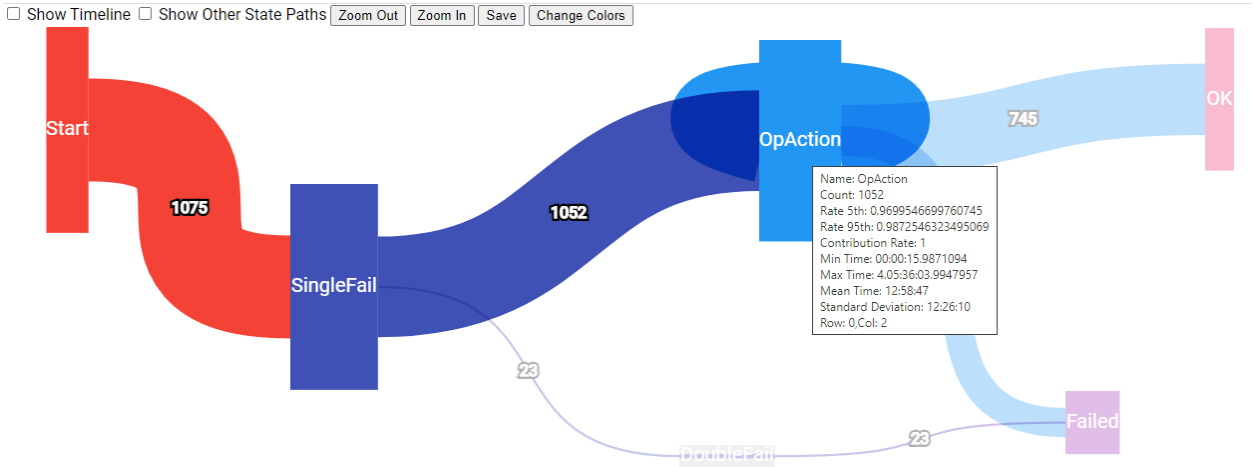


Figure 39. Simple Sankey result diagram with a hover over to show statistics.

The EMRALD Sankey visualization also offers a timeline-based graph, as shown in Figure 40, in which nodes are fixed along a horizontal axis such that the center of each node in the diagram corresponds to the mean time that state was entered. Distribution handles around each node show the standard deviation around the mean time. The idea of a time-based Sankey diagram is a new development for visualizing dynamic PRA results. While a standard Sankey diagram can show paths and order, the time and distribution data can be important in seeing relationships between events or critical narrow points in time.

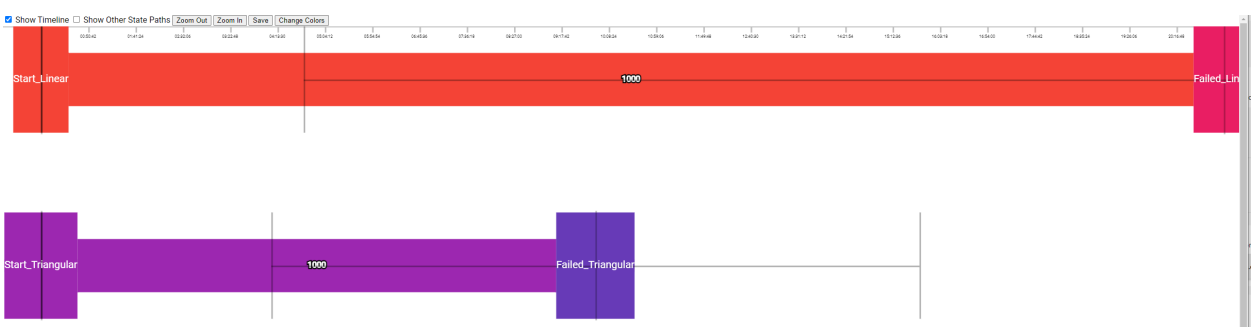


Figure 40. Time-based Sankey diagram showing the time and standard deviation whiskers.

The Sankey diagrams provides the following key benefits:

- Understand the flow of a model
- Quickly determine the largest paths causing the outcome
- Check and debug the model
- Compare variations of a model.

An industry collaborator, ARES Security, found the Sankey modeling very useful for their analysis and explaining outcomes. The diagram shown in Figure 41 is an example of their results for a fairly complicated model showing the attack and mitigation strategies for a fictional plant and scenario.

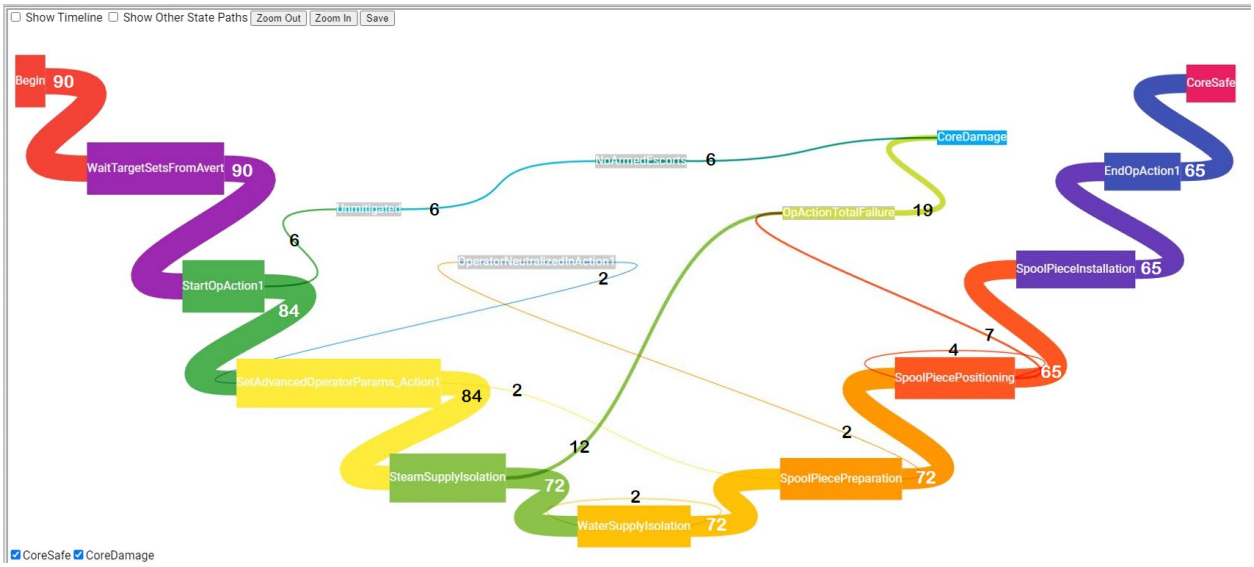


Figure 41. Sankey results from a complicated model.

4.2.3 GUI Development

The results GUI can be loaded from either the solver UI, shown in Figure 42, or the menu in the web-based modeling GUI. The GUI uses D3.js to create a scalable vector graphics (SVG)-based interface. In addition to the timeline view, the EMRALD Sankey tool is capable of handling loops and self-links, which are features required for visualizing the results but are not found in other Sankey diagram implementations. Within the GUI, the timeline view can be toggled on and off to allow viewing of a traditional Sankey diagram alongside the timeline along with the other top bar options shown in Figure 43. The colors of nodes are initially assigned automatically, but each node can be customized from a list of preset colors or colors the user inputs. Nodes can additionally be dragged freely in the default Sankey view or along the y-axis in the timeline view to improve readability. A list of key states along the bottom of the GUI allows the user to select which paths are displayed in the diagram. Unchecking a key state will hide states, links, and counts that led to that key state. Finally, the diagram can be saved from within the GUI to preserve the user-set colors and positions of nodes.

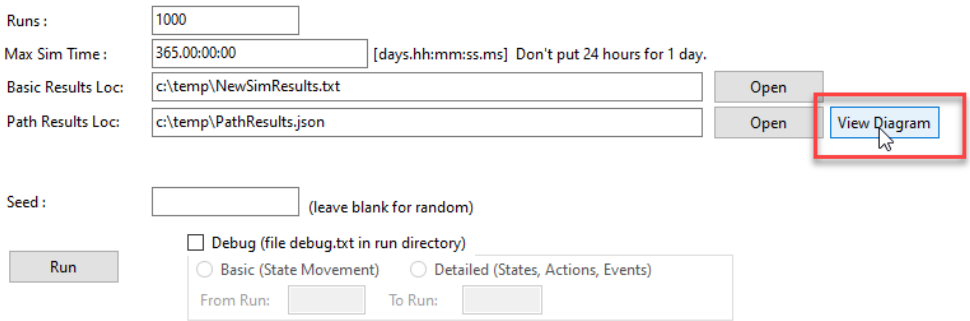


Figure 42. Option to open the Sankey results from the EMRALD solve engine GUI.

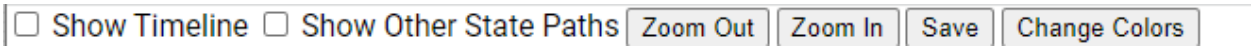


Figure 43. Options at the top of the Sankey result viewer.

5. REFERENCES

1. Parisi, C. et al. 2016. Demonstration of External Hazards Analysis. Report No. INL/EXT-16-39353. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/RiskInformed%20Safety%20Margin%20Characterization/Demonstration_of_External_Hazards_Analysis.pdf.
2. Coleman, J. et al. 2016. Multi-Hazard Advanced Seismic Probabilistic Risk Assessment Tools and Applications. Report No. INL/EXT-16- 40055. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/RiskInformed%20Safety%20Margin%20Characterization/Multi-Hazard_Advanced_Seismic_Probabilisitc_Risk_assessment_Tools_and_Applications.pdf.
3. Ma, Z., C. L. Smith, and S. R. Prescott. 2022. A Simulation-Based Dynamic Analysis Approach for Modeling Plant Response to Flooding Events. Report No. INL/EXT 17 40928, Rev. 1. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/RiskInformed%20Safety%20Margin%20Characterization/Simulation-basedDynamicApproachFlooding_RISA.pdf.
4. Parisi, C., et al. 2017. Risk-Informed External Hazards Analysis for Seismic and Flooding Phenomena for a Generic PWR. Report No. INL/EXT-17-42666. Idaho Falls, ID: Idaho National Laboratory, https://www.osti.gov/servlets/purl/1376899
5. Ma, Z., C. Smith, and S. Prescott. 2017. "A Simulation-Based Dynamic Approach for External Flooding Analysis in Nuclear Power Plants." In Proceedings of The 20th Pacific Basin Nuclear Conference, edited by H. Jiang, 55-69. Singapore: Springer. https://doi.org/10.1007/978-981-10-2311-8_6.
6. Energy I-Corps. n.d. “Teams.” Accessed August 22, 2023. https://energyicorps.inl.gov/SitePages/Teams.aspx.
7. Office of Technology Transitions. 2020. “Department of Energy Announces $33 Million for 2020 Technology Commercialization Fund Projects.” June 11, 2020. https://www.energy.gov/technologytransitions/articles/department-energy-announces-33-million-2020-technology.
8. Small Business Innovation Research. n.d. “Development of EMRALD Services in a Fully-Integrated RISMC Platform.” Accessed August 22, 2023, https://www.sbir.gov/node/1648965.
9. Christian, R., et al. 2020. Methodology and Application of Physical Security Effectiveness Based on Dynamic Force-on-Force Modeling. Report No. INL/EXT-20-59891. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/Physical%20Security/Methodology_Application_Physical_Effectiveness_based_on_FoF.pdf
10. Prescott, S., et al. 2023. Plant-Specific Model and Data Analysis using Dynamic Security Modeling and Simulation. Report No. INL/RPT-23-73490. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/Physical%20Security/Plant-SpecificModelDataAnalysisDynamicSecurity.pdf.
11. Park, J., et al. 2023. “Analysis of Tasks in Autonomous Systems Using the EMRALD Dynamic Risk Assessment Tool.” Paper presented at the 2023 International Conference on Applied Human Factors and Ergonomics, San Francisco, CA.
12. Park, J., et al. 2023. Simulation-Based Recovery Action Analysis Using the EMRALD Dynamic Risk Assessment Tool. Report No. INL/CON-23-71740.Idaho Falls, ID: Idaho National Laboratory, https://inldigitallibrary.inl.gov/sites/sti/sti/Sort_65630.pdf

13. Lew, R., et al. 2023. EMRALD-HUNTER: An Embedded Dynamic Human Reliability Analysis Module for Probabilistic Risk Assessment. Report No. INL/RPT-23-72783. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/RiskInformed%20Safety%20Margin%20Characterization/EMRALD-HUNTER.pdf.

14. Alhadhrami, S., et al. 2023. "Small Modular Reactor Condensate and Feedwater System Maintenance Methods by Utilizing Event Modeling Risk Assessment using Linked Diagram." In 13th Nuclear Plant Instrumentation, Control & Human-Machine Interface Technologies 1335-1344. doi.org/10.13182/NPICHMIT23-41354.

15. Wiltbank, N. E. and C. J. Palmer. 2021. "Dynamic PRA Prospects for the Nuclear Industry." Frontiers in Energy Research 9. https://doi.org/10.3389/fenrg.2021.750453.

16. Earthperson, A., et al. 2023. "A combined strategy for dynamic probabilistic risk assessment of fission battery designs using EMRALD and DEPM." Progress in Nuclear Energy 160 (June): 104673. https://doi.org/10.1016/j.pnucene.2023.104673.

17. MeV Summer School. n.d. "Welcome to MeV 2023." Accessed August 22, 2023. https://www.mevschool.net.

18. FPoliSolutions. n.d. "Products." Accessed August 22, 2023. https://fpolisolutions.com/products.

19. ARES. 2021. "ARES Security Corporation Awarded $1.1M DOE SBIR Phase 2 Grant." Last modified July 26, 2021. https://www.aressecuritycorp.com/perspectives/doe-sbir.

20. Centroid Lab. n.d. "Open 100 Coupled Simulation." Accessed August 28, 2023. https://centroidlab.com/project/open100-coupled-simulation.

21. Prescott, S., T. Wood, and M. Ziccarelli. 2022. Dynamic and Classical PRA Coupling using EMRALD and SAPHIRE. Report No. INL/RPT-22-70424. Idaho Falls, ID: Idaho National Laboratory, https://lwrs.inl.gov/RiskInformed%20Safety%20Margin%20Characterization/Dynamic_Classical_PRA.pdf

22. Lee, J. C. and N. J. McCormick. 2011. Risk and Safety Analysis of Nuclear Systems. Hoboken: Wiley.

23. Drouin, M., et al. 2013. Glossary of Risk-Related Terms in Support of Risk Informed Decisionmaking. Report No. NUREG-2122. Rockville, MD: U.S. Nuclear Regulatory Commission, https://www.nrc.gov/docs/ML1331/ML13311A353.pdf.

24. Prasad, M. H., et al. 2021. "Dynamic PSA Studies for Advanced Reactor Using RAVEN." Paper presented at the 2021 International Topical Meeting on Probabilistic Safety Assessment and Analysis, Virtual.

25. Bley, D. C., D. R. Buttemer, and J. W. Stetkar. 1988. "Light water reactor sequence timing: its significance to probabilistic safety assessment modeling." Reliability Engineering & System Safety 22 (1-4): 27-60. https://doi.org/10.1016/0951-8320(88)90066-X.

Appendix A

This section presents the derivations of the results used in Equations (12)) and (13)). The process followed to derive the availability equations is given in the textbook, *Risk Analysis and Safety of Nuclear Systems* [1].

A-1. Derivations of Availability Equations

A-1.1 Two Components Fail in Series, One Repairperson Available

$$Failure\ Rate: \lambda, Repair\ Rate: \mu, Mission\ Time: t \tag{14}$$

$$M = \begin{matrix} -2\lambda & \mu & 0 \\ 2\lambda & -(\mu+\lambda) & \mu \\ 0 & \lambda & -\mu \end{matrix} \tag{15}$$

$$SI - M = \begin{matrix} s+2\lambda & -\mu & 0 \\ -2\lambda & (\mu+\lambda) & -\mu \\ 0 & -\lambda & s+\mu \end{matrix} \tag{16}$$

$$SI - M^T = \begin{matrix} s+2\lambda & -2\lambda & 0 \\ -\mu & (\mu+\lambda) & -\lambda \\ 0 & -\mu & s+\mu \end{matrix} \tag{17}$$

$$\Delta = 2\lambda^2 s + 3\lambda s^2 + 2\lambda s\mu + s^3 + 2s^2\mu + s\mu^2 + \mu^2 \tag{18}$$

$$s = 0;\ s_1 = -0.5\left[(3\lambda + 2\mu) + \sqrt{\lambda^2 + 4\lambda\mu}\right];\ s_2 = -0.5[(3\lambda + 2\mu) - \sqrt{\lambda^2 + 4\lambda\mu}] \tag{19}$$

$$A(s) = P_1(s) \tag{20}$$

$$A(s) = \frac{s^2 + s(2\mu + \lambda) + \mu^2}{s(s - s_1)(s - s_2)} \tag{21}$$

$$A(t) = L^{-1}A(s) \tag{22}$$

$$A(t) = \frac{e^{s_1 t}(s_1(2\mu + \lambda) + s_1^2 + \mu^2)}{s_1(s_1 - s_2)} + \frac{e^{s_2 t}(s_2(2\mu + \lambda) + s_2^2 + \mu^2)}{s_1(s_1 - s_2)} + \frac{\mu^2}{s_1 s_1} \tag{23}$$

A-1.2 Two Components Fail in Series, Two Repairpersons Available

$$Failure\ Rate: \lambda, Repair\ Rate: \mu, Mission\ Time: t \tag{24}$$

$$M = \begin{matrix} -2\lambda & \mu & 0 \\ 2\lambda & -(\mu+\lambda) & 2\,\mu \\ 0 & \lambda & -2\mu \end{matrix} \tag{25}$$

$$SI - M = \begin{matrix} s+2\lambda & -\mu & 0 \\ -2\lambda & (\mu+\lambda) & -2\mu \\ 0 & -\lambda & s+2\mu \end{matrix} \tag{26}$$

$$SI - M^T = \begin{matrix} s+2\lambda & -2\lambda & 0 \\ -\mu & (\mu+\lambda) & -\lambda \\ 0 & -2\mu & s+2\mu \end{matrix} \tag{27}$$

$$\Delta = 2\lambda^2 s + 3\lambda s^2 + 4\lambda s\mu + s^3 + 3s^2\mu + 2s\mu^2 \tag{28}$$

$$s = 0;\ s_1 = -2(\lambda+\mu);\ s_2 = -(\lambda+\mu) \tag{29}$$

$$A(s) = P_1(s) \tag{30}$$

$$A(s) = \frac{s^2 + s(3\mu+\lambda) + 2\mu^2}{s(s-s_1)(s-s_2)} \tag{31}$$

$$A(t) = L^{-1}A(s) \tag{32}$$

$$A(t) = \frac{e^{s_1 t}(s_1(3\mu+\lambda) + s_1^2 + 2\mu^2)}{s_1(s_1-s_2)} + \frac{e^{s_2 t}(s_2(3\mu+\lambda) + s_2^2 + 2\mu^2)}{s_1(s_1-s_2)} + \frac{2\mu^2}{s_1 s_1} \tag{33}$$

A-1.3 Reference

1. Lee, J. C. and N. J. McCormick. 2011. *Risk and Safety Analysis of Nuclear Systems*. Hoboken: Wiley.