INL-RPT-23-73391
Revision 1

Resilience Development for Electric Energy Delivery Systems (ResDEEDS)

July 2023

A Tool for Power System Resilience Planning

Megan Culler
Idaho National Laboratory

Mathew L. Wymore
Iowa State University

Michael Cutshaw
Idaho National Laboratory

Zachary Priest
Idaho National Laboratory

Jody Dillon
Energy Reform

Manuel Marin
Energy Reform

Idaho National Laboratory

*INL is a U.S. Department of Energy National Laboratory
operated by Battelle Energy Alliance, LLC*

DISCLAIMER

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

INL-RPT-23-73391
Revision 1

Resilience Development for Electric Energy Delivery Systems (ResDEEDS)

A Tool for Power System Resilience Planning

Megan Culler, *Idaho National Laboratory*
Mathew L. Wymore, *Iowa State University*
Michael Cutshaw, *Idaho National Laboratory*
Zachary* Priest, *Idaho National Laboratory
Jody Dillon, *Energy Reform*
Manuel Marin, *Energy Reform*

July 2023

Idaho National Laboratory
Critical Infrastructure Security
Idaho Falls, Idaho 83415

http://www.inl.gov

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

Page intentionally left blank

CONTENTS

FIGURES

Page intentionally left blank

1. INTRODUCTION

Reliability planning is a well-established practice to balance generation and load in the presence of scheduled and unscheduled outages, ensuring that voltage and frequency are maintained at nominal levels. The practice of resilience planning is not well-established, or even well-defined. Recent research has aimed to support resilient operation of power systems through better preparedness and recover plans for when hazard events occur, specific upgrades to systems to enhance resilience, and control methods and operational reactions to hazard events [1, 2, 3]. However, few methods exist to guide more generic resilience planning that is not focused on optimizing a particular technology or control strategy. The planning stage of the Resilience Framework for Electric Energy Delivery Systems, developed by Idaho National Laboratory (INL) aims to accomplish this [4]. This framework aligns with the perspective put forth by the National Academies of Sciences, Engineering, and Medicine,

"An all-hazards approach to resilience planning is essential, but, with the exception of a few general strategies, there is no "one-size-fits-all" solution to planning for and recovering from major outages. The notion of resilience has to address multiple types of events and operate in a system with multiple overlapping institutions, service providers, grid configurations, ownership structures, and regulatory systems" [2].

To that end, the INL Resilience Framework is built to be customized to a particular system's characteristics, resilience goals, and hazards that it is likely to face. The challenge with a framework that is built to be customized is that it requires more effort from the user to understand the framework and apply it correctly to a particular system. The goal of the web application described here is to automate the framework application as much as possible. Inputs from users are standardized into a common format, but the modeling platform used is a modular framework built to adapt to any system. Baseline effects from common hazards are programmed with stochastic variables to allow for quick exploration of hazard impacts, but can be customized down to component-level impact if a user desires a high level of detail and specificity.

1.1 Resilience Framework for Electric Energy Delivery Systems

The INL Resilience Framework was developed to provide a process for resilience planning, operation, and evaluation throughout a system's lifecycle [4]. The Resilience Framework provides a methodology for resilience planning that involves identifying key system characteristics, resilience goals, resilience hazards, and performing simulations to see how the system performs against the hazards The application presented here focuses on the planning stage of the framework, which contains seven steps.

1. Identify system qualities
2. Define system resilience goals and metrics.
3. Prioritize physical and cyber hazards.
4. Model hazard.
5. Prioritize hazards and risk mitigations.
6. Evaluate against all business risks.
7. Implement changes and operate system.

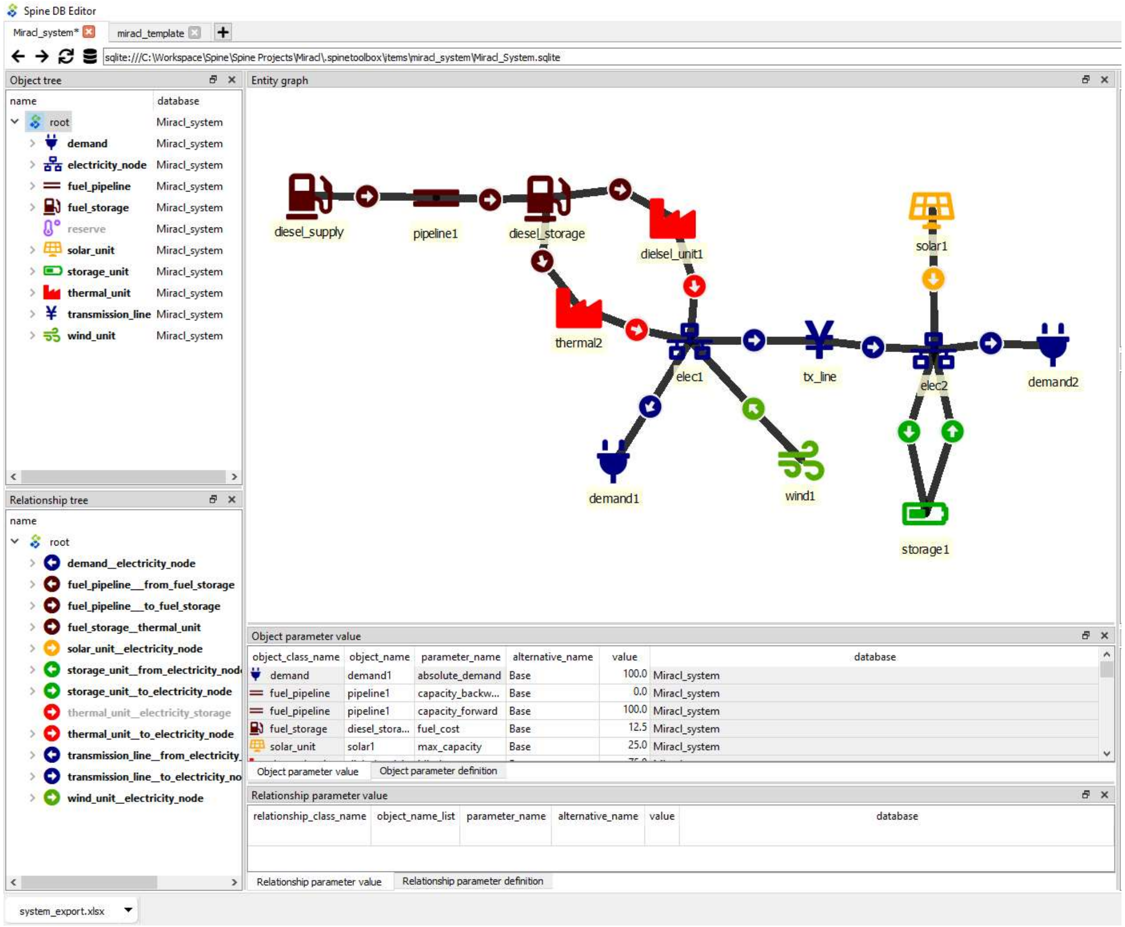
By using this framework for resilience planning, system planners and operators can compare various investment options, configurations, or operational responses' performance against hazards of interest and get qualitative data to support a decision based on resilience.

1.2 Spine Toolbox

The Spine Toolbox is an application that provides means to define, manage, and execute complex data processing and computational tasks, such as modeling energy systems [5]. It is defined to be highly modular and system agnostic, using basic building blocks and databases to create models and simulation parameters for complex systems.

Unlike other power system modeling platforms, Spine is built to accommodate new functionalities, allowing users to determine the timescales, level of detail, and optimization goals for their application.

Spine uses workflows to represent modelling exercises. A Spine workflow is built as a directed acyclic graph (DAG) where vertices denote computing tasks and edges denote flow of information. At its core, Spine models are built of nodes and connections. Constraints and properties of the nodes and connections can be defined to create a context-informed system model. For example, a thermal generation node can be defined by an idle heat rate, incremental heat rate, max capacity, ramp rates, fuel efficiency and more. A transmission line connection can be described by its forward capacity, backwards capacity, resistance, and reactance. Figure 1 shows an example of a power system that includes diesel, wind, and solar generation, a battery storage unit, two primary loads, and a transmission line between the load centers.



sqlite:///C:\Workspace\Spine\Spine Projects\Miracl\.spinetoolbox\items\miracl_system\Miracl_System.sqlite

Object tree

name	database
root	Miracl_system
demand	Miracl_system
electricity_node	Miracl_system
fuel_pipeline	Miracl_system
fuel_storage	Miracl_system
reserve	Miracl_system
solar_unit	Miracl_system
storage_unit	Miracl_system
thermal_unit	Miracl_system
transmission_line	Miracl_system
wind_unit	Miracl_system

Relationship tree

name
- root
  - demand__electricity_node
  - fuel_pipeline__from_fuel_storage
  - fuel_pipeline__to_fuel_storage
  - fuel_storage__thermal_unit
  - solar_unit__electricity_node
  - storage_unit__from_electricity_nod
  - storage_unit__to_electricity_node
  - thermal_unit__electricity_storage
  - thermal_unit__to_electricity_node
  - transmission_line__from_electricity_
  - transmission_line__to_electricity_no
  - wind_unit__electricity_node

Object parameter value

object_class_name	object_name	parameter_name	alternative_name	value	database
demand	demand1	absolute_demand	Base	100.0	Miracl_system
fuel_pipeline	pipeline1	capacity_backw...	Base	0.0	Miracl_system
fuel_pipeline	pipeline1	capacity_forward	Base	100.0	Miracl_system
fuel_storage	diesel_stora...	fuel_cost	Base	12.5	Miracl_system
solar_unit	solar1	max_capacity	Base	25.0	Miracl_system

Object parameter value | Object parameter definition

Relationship parameter value

relationship_class_name	object_name_list	parameter_name	alternative_name	value	database

Relationship parameter value | Relationship parameter definition

system_export.xlsx

Figure 1: Example Spine database editor displaying a power system model.

The output of each computing task is fed as an input to all their successors, and so on. In this way, a complex modelling task is split into several simpler units of computation which simplifies the job of the modeler by allowing them to reason about one of such units at a time.

1.3 Resilience Application

The planning process in the Resilience Framework was shown to be effective at measuring resilience of a system in response to certain system conditions that form a hazard. Two demonstrations with real power system data showed how the quantitative outputs from this analysis could be used to support financial decisions about investments in the system.

The purpose of this application is to automate the planning stage of the Resilience Framework. It allows users to upload data from their system, define relationships, and evaluate the system's performance against various hazards.

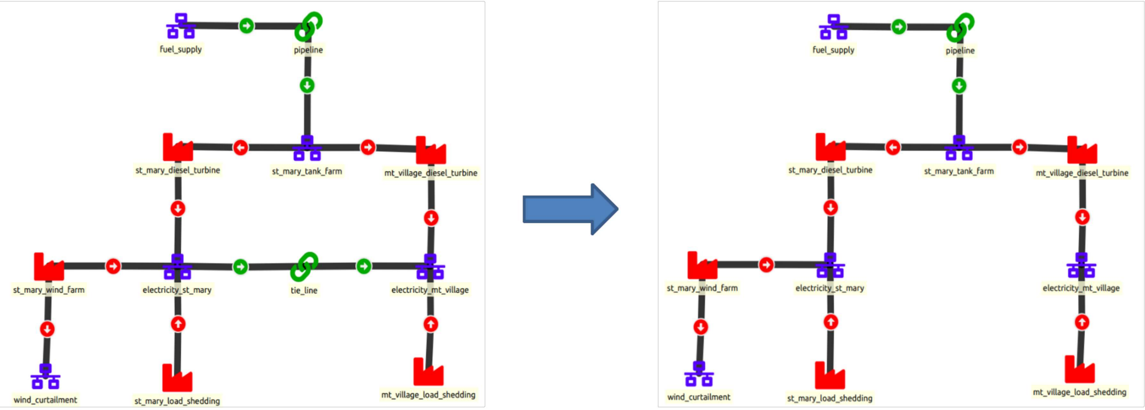
A key feature of the application is the built-in hazard models, which can be customized by the user to reflect exactly the type of event that they wish to analyze. However, there remains a stochastic component to the hazards, which means that the user does not have to define precisely the impacts of a mock hazard, but more generally analyze what types of impacts are expected, especially if the hazard simulation is run multiple times to bound the results.

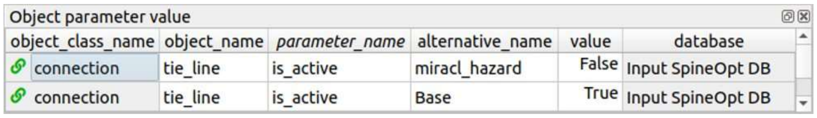
2. Hazard Modeling

Hazards in Spine are modeled as changes to the system parameters, such as the capacity of a generator or available flow on a line. Hazard impacts are calculated by running the same simulation in two conditions: one as a "base case" with all components behaving normally, and one as a hazard with the changes in place. The impacts are calculated as the difference in tracked metrics between the two scenarios.

2.1.1 Scenarios

Scenarios describe the conditions of the system for a simulation run of the system. Scenarios are made of up a collection of "alternatives," which describe the state of particular components during a hazard. For example, a tornado may cause damage to distribution feeders and temporary derating of wind turbines as they are shut down to avoid damage from overspeed. This would be represented as alternatives that include changing the capacity of certain lines to zero, and reducing, or entirely eliminating, wind power production for a period of time during the simulation. Figure 2 shows a base case system and a system with the hazard applied. The tie line connecting the two electric systems has an alternative defined which takes the line out of service. The hazard scenario is defined by this alternative, but other alternatives could be added to enhance the complexity of the scenario.





Object parameter value

object_class_name	object_name	*parameter_name*	alternative_name	value	database
connection	tie_line	is_active	miracl_hazard	False	Input SpineOpt DB
connection	tie_line	is_active	Base	True	Input SpineOpt DB

Figure 2: A base case system and a hazard scenario for this system with one alternative, the tie line out of service.

[Figure with one or multiple sub-figures that highlights alternatives for components and collections of alternatives as scenarios]

2.1.2 Timing

Each hazard is assigned a start time and a duration where it could occur. Because all parameters remain the same outside of the hazard duration, the simulation only needs to be run for the duration period in order to evaluate the resilience metrics and compare results between the hazard scenarios and bases cases. The exception to this would be if a user wishes to evaluate a net effect of multiple hazards over a longer period of time, for example, a fuel shortage event in the winter that reduces reserves, making the system less prepared to handle a summer load peaking event that produces record demand. To evaluate the resilience of the system against different types of hazards independently, which we believe would be more common and practical, multiple hazards can be defined, and separate iterations of SpineOpt will be run for the base case and hazard case of each independent hazard.

2.1.3 Stochastic variables

The user interface allows for definition of exactly which components are impacted by a hazard and how they are impacted (whether they are derated, eliminated, have a new time series of data associated with them, etc.). However, one of the challenges with resilience planning is that it is difficult to predict exactly how a hazard, such as a hurricane, winter storm, or cyberattack, might impact a system. Simulations also require assigning a time that the hazard starts and how long it lasts, but this too may be difficult to predict.

To combat that challenge, the resilience application allows for a level of randomness to be added to the hazards. The first layer is randomizing the start time of the hazard. Given a simulation start time, the hazard may start immediately, or it may fall randomly within a range of possible start times using a mean time to "start of hazard."

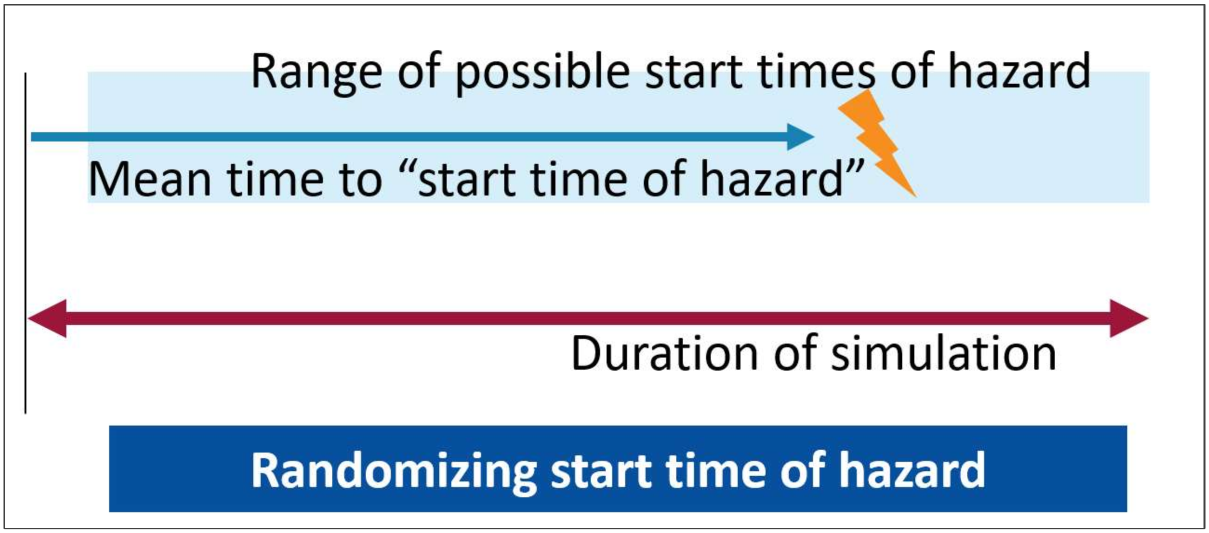

Range of possible start times of hazard
Mean time to "start time of hazard"
Duration of simulation
Randomizing start time of hazard

Figure 3: Randomizing start time of hazard from simulation start time.

The next layer of randomization that can be added is selecting which components are impacted. For example, we might say that 15% of all generating units are affected, or 30% of all lines are affected. Applying this randomization allows the SpineOpt trial to apply the effects to specific components. Using this approach as is carries some risk. For example, a tornado path is not going to randomly affect lines throughout a system but will instead affect lines based on a geographic path. While this level of modeling is possible to add in future iterations of this application, it is not an existing functionality. If the user has the knowledge to describe exactly which components are affected by their hazard of interest, it may be more effective to define these. However, if the user wants a general picture of the types of impacts, or if the user is interested in broadly comparing the resilience of different system configurations or future upgrades, then the random approach may work well.

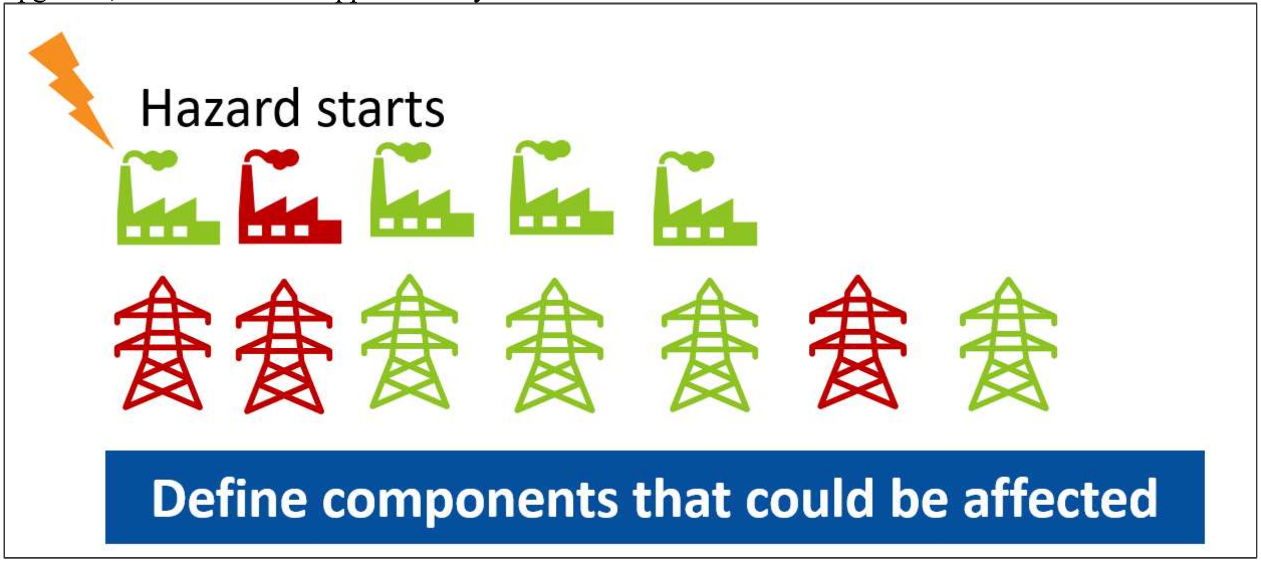


Figure 4: Randomly selecting which particular components are affected based on hazard definition.

2.1.4 Future Work

There are other layers of hazard modeling that have been discussed for the application but have not yet been implemented. These include:

- Randomizing the failure interval and using a mean time to repair so that not every component may fail and be repaired at the same time. This would add a level of realism to the simulation.
- Using different probabilistic distributions (normal, exponential, uniform, etc.) to describe different stochastic variables.
- Randomizing the amount of degradation that a component experiences.
- Multiple trials: With each layer of randomization that is added, more variables are at play. By running the simulation multiple times, we can get an average picture of what types of impacts are experienced during a certain hazard. In this way, the random selections that form a detailed hazard are extrapolated to general effects.

3. User interface

3.1 Overview

The user interacts with the application via a web interface built using the Flask web framework for Python and the Jinja templating engine. The web interface supports Okta authentication for centralized deployments; it can also be run locally using Flask's built-in web server. The web interface workflow consists of seven pages, each of which supports one or more steps of the planning stage of the INL Resilience Framework. The pages form an iterative workflow that allows a user to input their baseline system information, hazard definitions, and modeling parameters. The user then prioritizes hazards and defines goals for various resilience metrics for each hazard. The web interface then calls the Spine Toolbox to evaluate the system on these metrics. The user is given results for the metrics through the web interface, at which point the user can iterate on the system until it meets the resilience goals. Once the user is satisfied with the system, they can view a page that summarizes the differences between the initial baseline system and the final system that was evaluated.

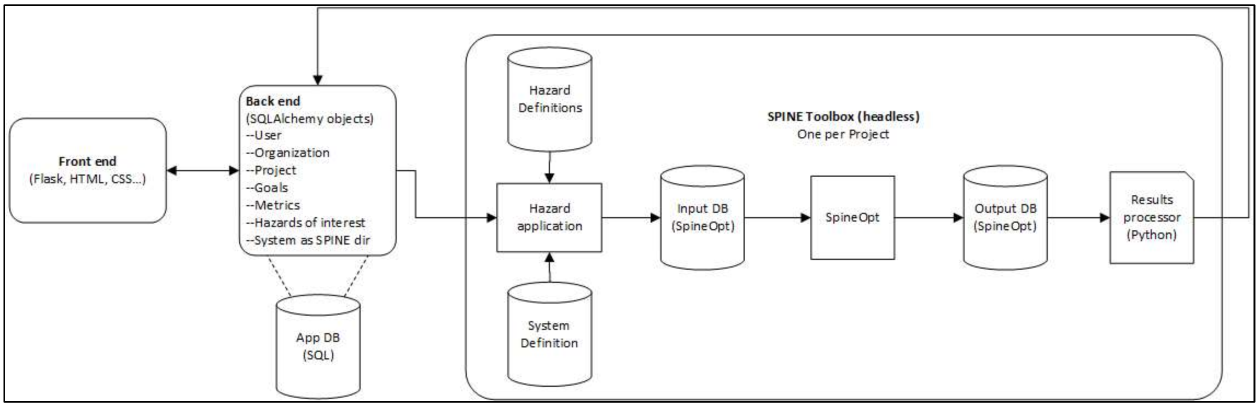
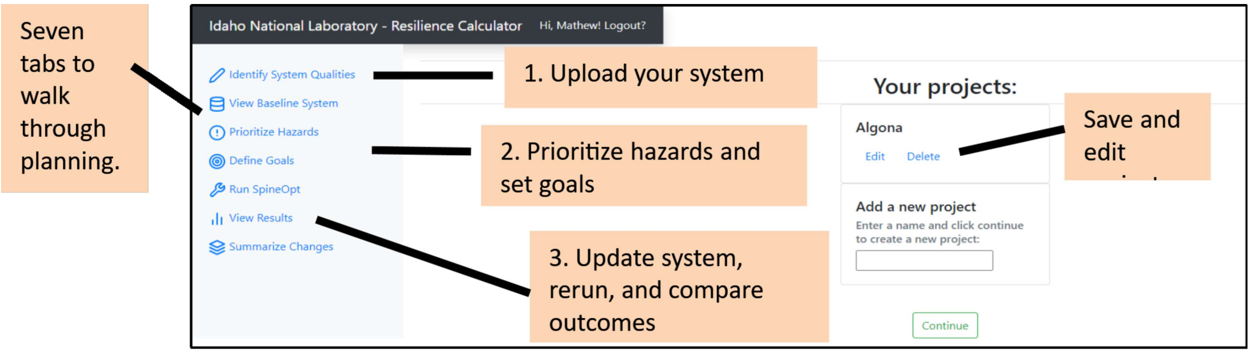


Figure 5: Application workflow.

3.2 Projects

We define a project as a system being evaluated, and the hazards and goals associated with that evaluation. After logging in (if authentication is enabled), the user is presented with a list of their projects. The user can choose to continue or delete a previous projects, or create a new project.



Enter a name and click continue to create a new project:

Figure 6: The project selection page.

3.3 Interaction with Spine

The application relies on Spine for quantitative evaluation of the projects. Each project corresponds to a unique directory that contains the Spine Toolbox files used in the Spine simulations for that project. This directory is automatically built from a template when the project is created.

The web interface interacts with Spine in two ways. First, to run the Spine Toolbox workflow (or select components of it), the Flask application uses Spine Toolbox's command-line interface via Python's subprocess module. The Flask application abstracts this method of interaction with a SpineToolbox class.

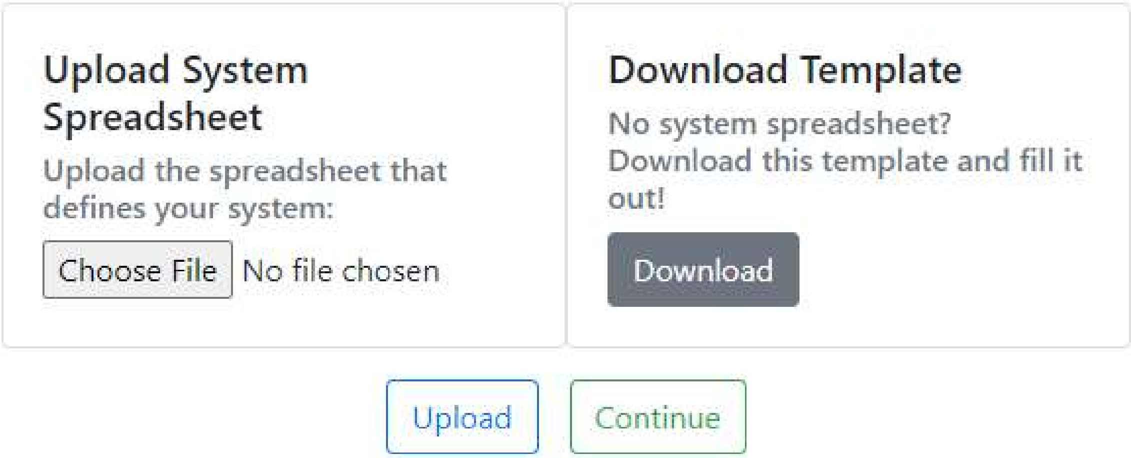
FIGURE OF SPINE INTERACTION/CLASSES

The second method of interaction between the web application and Spine is the SpineDB API. This is an open-source Python package for reading/writing databases in the SpineDB format. The web application uses SpineDB API any time data is read from the Spine project (such as when retrieving calculated resilience metric values) or any time SpineDB objects or relationships are modified through a web form. The web application abstracts its use of SpineDB API through a SpineDBSession class and supporting SpineObject, SpineRelationship, and SpineScenario classes.

3.4 Web Interface Workflow

3.4.1 Identify System Qualities

After creating a new project, users are presented with the Identify System Qualities page. On this page, the user can upload the spreadsheet that defines their system. This system is imported into Spine and saved as the *baseline system***.** The baseline system acts as the point of comparison when later determining what modifications need to be made to a system in order to meet resilience goals. The baseline system remains constant unless the user returns to this page and uploads a new spreadsheet, which effectively restarts the project.



Upload System Spreadsheet

Upload the spreadsheet that defines your system:

Download Template

No system spreadsheet? Download this template and fill it out!

Figure 7: The Identify System Qualities page.

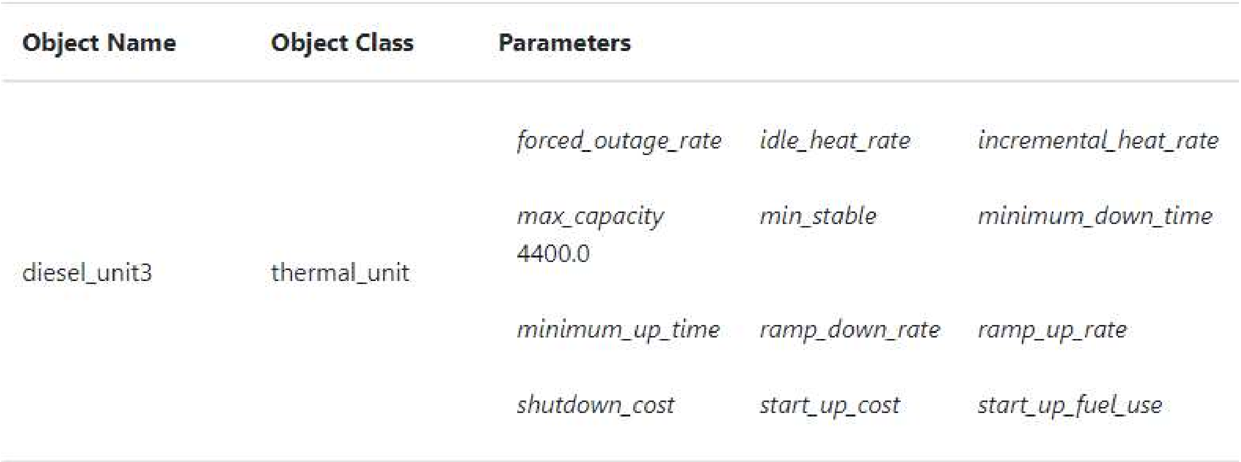
The spreadsheet uploaded on this page also contains definitions of hazards, as described above, and modeling parameters that are passed to Spine. These parameters, as well as the system format, are defined in a template spreadsheet, which is available for download on this page.

When the spreadsheet is uploaded, the Spine Toolbox workflow is run for the baseline system, and the results are saved as the baseline metric values. After the Spine Toolbox command has completed, the output of the command is shown at the bottom of the page for informational purposes.

3.4.2 View Baseline System

After uploading a spreadsheet, the user clicks Continue to navigate to the View Baseline System page. This page shows the data from the spreadsheet, as imported into and read back out of Spine. This page is read-only and allows the user to confirm that their system was imported correctly. Additionally, the user can always navigate back to this page to see what their baseline system looks like.

The data is shown in two tables. The first table has a row for each Spine object (e.g. a generator), and all parameters for that object are shown, including parameters with no explicit value. The second table shows Spine relationships (e.g. transmission lines) and the objects they connect.

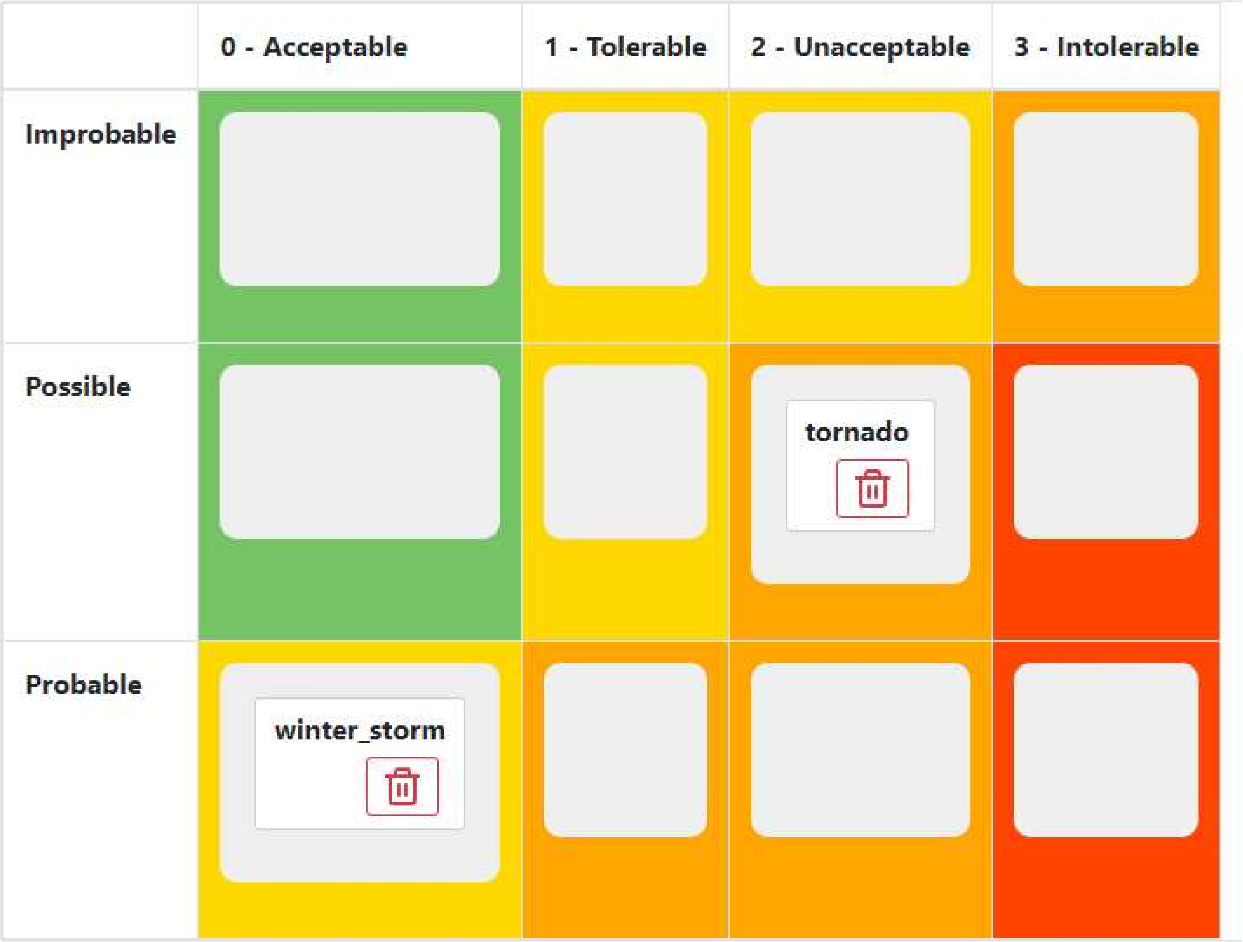


Object Name	Object Class	Parameters		
diesel_unit3	thermal_unit	*forced_outage_rate*	*idle_heat_rate*	*incremental_heat_rate*
		max_capacity 4400.0	*min_stable*	*minimum_down_time*
		minimum_up_time	*ramp_down_rate*	*ramp_up_rate*
		shutdown_cost	*start_up_cost*	*start_up_fuel_use*

Figure 8: Object data on the View Baseline System page. An example thermal unit named diesel_unit3 is shown, with only the max_capacity parameter explicitly set.

3.4.3 Prioritize Hazards

On the Prioritize Hazards page, users drag-and-drop hazards (imported into Spine from the uploaded spreadsheet) onto a risk matrix. This risk matrix assigns a risk-based color coding to each hazard, which is used on the remaining pages to sort the hazards and help the user organize their priorities.

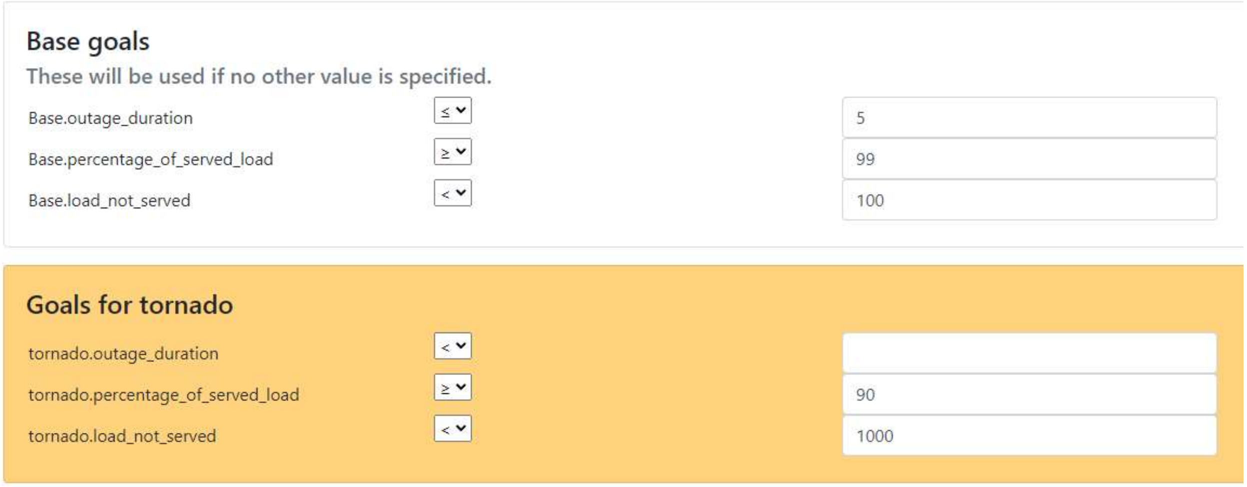


	0 - Acceptable	1 - Tolerable	2 - Unacceptable	3 - Intolerable
Improbable	🟩	🟨	🟨	🟧
Possible	🟩	🟨	🟧 tornado	🟥
Probable	🟨 winter_storm	🟧	🟧	🟥

Figure 9: Risk matrix on the Prioritize Hazards page.

3.4.4 Define Goals

After prioritizing hazards, the users are directed to the Define Goals page. On this page, they can create quantitative resilience goals for their system. Users are allowed to specify a target value, as well as the desired comparison operator, for each metric calculated by the Spine simulations. Goals can be specified individually for each hazard; for example, a user may want to achieve a very low amount of load not served for a winter storm, but may decide to tolerate a larger amount of load not served in the case of a tornado. The user can also set *base goals*, which are goals that should be met when no hazard is present. The base goal values are used for any hazard-specific goals that are not specified, reducing the number of goals that a user needs to explicitly set.



Base goals

These will be used if no other value is specified.

Base.outage_duration	≤	5
Base.percentage_of_served_load	≥	99
Base.load_not_served	<	100

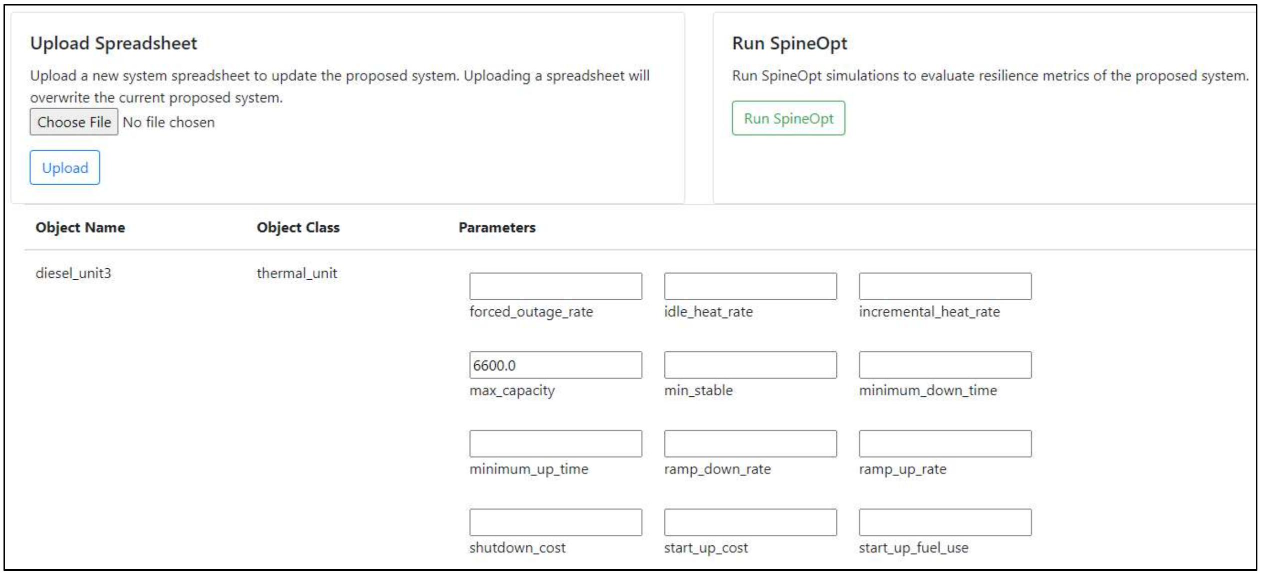
Goals for tornado

tornado.outage_duration	<	
tornado.percentage_of_served_load	≥	90
tornado.load_not_served	<	1000

Figure 10: Base goals and hazard-specific goals on the Define Goals page.

3.4.5 Run SpineOpt

The Run SpineOpt page is the core of the iterative workflow of the web interface. On this page, users modify their system and run the Spine Toolbox workflow on it. The modified system is saved as the *proposed system* (distinct from the baseline system). The proposed system can be modified in two ways: the user can update the parameter fields for the Spine objects in a web form, or the user can upload a new system spreadsheet. Both methods update the values in Spine. Uploading a new spreadsheet wipes out any changes made via the former method.



Upload Spreadsheet

Upload a new system spreadsheet to update the proposed system. Uploading a spreadsheet will overwrite the current proposed system.

Choose File No file chosen

Upload

Run SpineOpt

Run SpineOpt simulations to evaluate resilience metrics of the proposed system.

Run SpineOpt

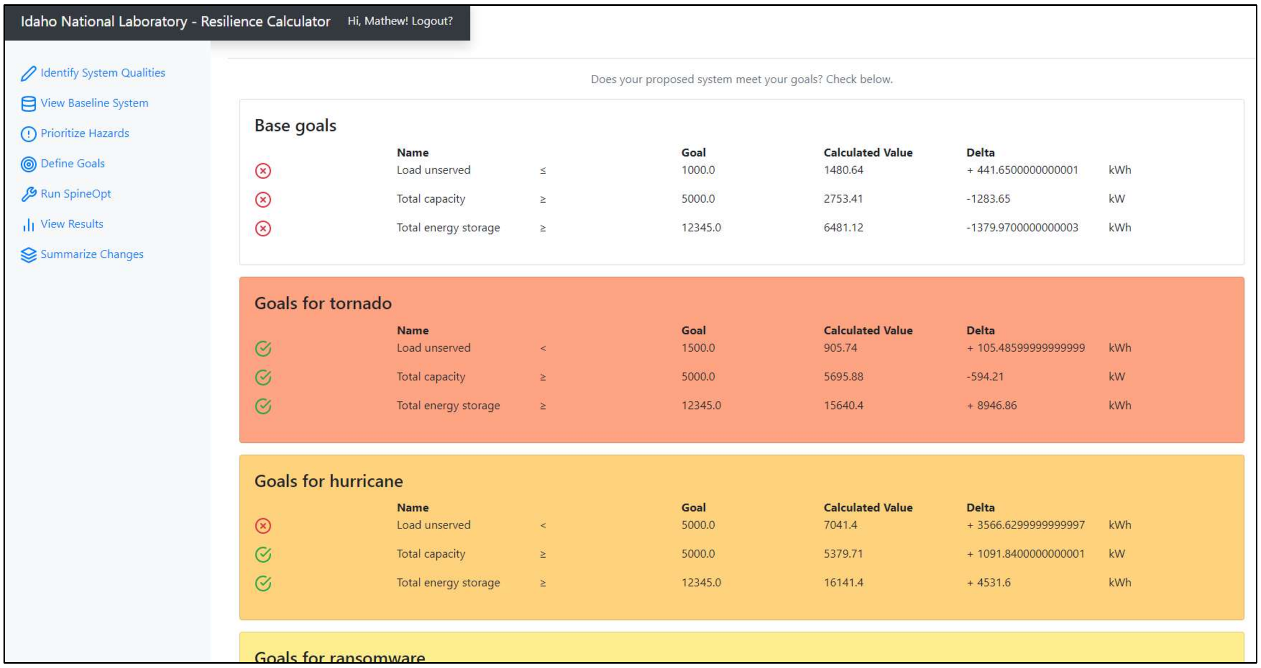
Object Name	Object Class	Parameters
diesel_unit3	thermal_unit	forced_outage_rate: idle_heat_rate: incremental_heat_rate: max_capacity: 6600.0 min_stable: minimum_down_time: minimum_up_time: ramp_down_rate: ramp_up_rate: shutdown_cost: start_up_cost: start_up_fuel_use:

Figure 11: The Run SpineOpt page.

After making modifications to the proposed system, the user can click Run SpineOpt to start the SpineOpt simulations. This runs simulations for each hazard, as described above, and calculates the programmed set of quantitative resilience metrics. When the simulations complete, the user is redirected to the View Results page.

3.4.6 View Results

On the View Results page, the user is shown the values of the resilience metrics calculated by the SpineOpt simulations for each hazard, as well as for the base (no hazard) case. An icon is used to visually indicate whether or not each goal was met. The *delta* (difference between the calculated value and the goal) is also given. From this page, the user can either choose to return to the Run SpineOpt page and further modify their proposed system, or continue on to the Summarize Changes page.



Does your proposed system meet your goals? Check below.

Base goals

	Name		Goal	Calculated Value	Delta	
✗	Load unserved	≤	1000.0	1480.64	+ 441.6500000000001	kWh
✗	Total capacity	≥	5000.0	2753.41	-1283.65	kW
✗	Total energy storage	≥	12345.0	6481.12	-1379.9700000000003	kWh

Goals for tornado

	Name		Goal	Calculated Value	Delta	
✓	Load unserved	<	1500.0	905.74	+ 105.48599999999999	kWh
✓	Total capacity	≥	5000.0	5695.88	-594.21	kW
✓	Total energy storage	≥	12345.0	15640.4	+ 8946.86	kWh

Goals for hurricane

	Name		Goal	Calculated Value	Delta	
✗	Load unserved	<	5000.0	7041.4	+ 3566.6299999999997	kWh
✓	Total capacity	≥	5000.0	5379.71	+ 1091.8400000000001	kW
✓	Total energy storage	≥	12345.0	16141.4	+ 4531.6	kWh

Goals for ransomware

Figure 12: The View Results page.

3.4.7 Summarize Changes

The final page, Summarize Changes, shows the user the difference between their proposed system and their baseline system. This is intended to help the user keep track of the changes they are proposing to implement in order to reach their resilience goals. The summary is shown in three tables: objects that have been added to the system, objects that have been removed from the system, and objects that have been changed (i.e., objects with modified parameters). Clicking Finish on this page returns the user to the project selection page.