



**Calhoun: The NPS Institutional Archive**  
**DSpace Repository**

---

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

---

2022-12

# MICROGRID RESILIENCE ANALYSIS SOFTWARE DEVELOPMENT

Buetow, Thomas M.; Ingalls, James D.; Ledden, Michael J.,  
Jr.; Palmer, Andrew R.; Perez, Ricardo

Monterey, CA; Naval Postgraduate School

---

<https://hdl.handle.net/10945/71596>

---

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States

*Downloaded from NPS Archive: Calhoun*



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

**Dudley Knox Library / Naval Postgraduate School**  
**411 Dyer Road / 1 University Circle**  
**Monterey, California USA 93943**

<http://www.nps.edu/library>



**NAVAL  
POSTGRADUATE  
SCHOOL**

**MONTEREY, CALIFORNIA**

**SYSTEMS ENGINEERING  
CAPSTONE REPORT**

**MICROGRID RESILIENCE ANALYSIS  
SOFTWARE DEVELOPMENT**

by

Thomas M. Buetow, James D. Ingalls, Michael J. Ledden Jr.,  
Andrew R. Palmer, and Ricardo Perez

December 2022

Advisor:

Douglas L. Van Bossuyt

Co-Advisors:

Ronald E. Giachetti, Giovanna Oriti,  
Daniel Reich, Mark M. Rhoades  
and Corina L. White

**Approved for public release. Distribution is unlimited.**

THIS PAGE INTENTIONALLY LEFT BLANK

<b>REPORT DOCUMENTATION PAGE</b>			<i>Form Approved OMB No. 0704-0188</i>
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC, 20503.			
<b>1. AGENCY USE ONLY (Leave blank)</b>	<b>2. REPORT DATE</b> December 2022	<b>3. REPORT TYPE AND DATES COVERED</b> Systems Engineering Capstone Report	
<b>4. TITLE AND SUBTITLE</b> MICROGRID RESILIENCE ANALYSIS SOFTWARE DEVELOPMENT			<b>5. FUNDING NUMBERS</b>
<b>6. AUTHOR(S)</b> Thomas M. Buetow, James D. Ingalls, Michael J. Ledden Jr., Andrew R. Palmer, and Ricardo Perez			
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> Naval Postgraduate School Monterey, CA 93943-5000			<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b> N/A			<b>10. SPONSORING / MONITORING AGENCY REPORT NUMBER</b>
<b>11. SUPPLEMENTARY NOTES</b> The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
<b>12a. DISTRIBUTION / AVAILABILITY STATEMENT</b> Approved for public release. Distribution is unlimited.			<b>12b. DISTRIBUTION CODE</b> A
<b>13. ABSTRACT (maximum 200 words)</b>  Military installation microgrids need to be resilient to a variety of potential disruptions (storms, attacks, et cetera). Various metrics for assessing microgrid resilience have been described in literature, and multiple tools for simulating microgrid performance have been constructed; however, it is often left to system owners and maintainers to bring these efforts together to identify and realize effective, efficient improvement strategies. Military microgrid stakeholders have expressed a desire for an integrated, unified platform that provides these multiple capabilities in a coordinated fashion. In support of these endeavors, analysis methods developed by NPS and NAVFAC Expeditionary Warfare Center researchers for measuring microgrid resilience have been integrated into an existing web-based microgrid power flow simulation and distributed energy resource rightsizing software tool. This was achieved by the development of additional functions and methods within the existing software platform code base, and expansion of the application programming interface (API). These API additions enabled access to the new calculation and analysis capabilities, as well as increased control over power flow simulation parameters. These analytical and functional contributions were validated through a design of experiments, including comparison to independently generated data, and factorial analysis.			
<b>14. SUBJECT TERMS</b> microgrid, resilience, systems engineering, MBSE			<b>15. NUMBER OF PAGES</b> 157
			<b>16. PRICE CODE</b>
<b>17. SECURITY CLASSIFICATION OF REPORT</b> Unclassified	<b>18. SECURITY CLASSIFICATION OF THIS PAGE</b> Unclassified	<b>19. SECURITY CLASSIFICATION OF ABSTRACT</b> Unclassified	<b>20. LIMITATION OF ABSTRACT</b> UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)  
Prescribed by ANSI Std. Z39-18

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release. Distribution is unlimited.**

**MICROGRID RESILIENCE ANALYSIS SOFTWARE DEVELOPMENT**

Thomas M. Buetow, James D. Ingalls, Michael J. Ledden Jr.,  
Andrew R. Palmer, and Ricardo Perez

Submitted in partial fulfillment of the  
requirements for the degree of

**MASTER OF SCIENCE IN SYSTEMS ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL  
December 2022**

Lead Editor: James D. Ingalls

Reviewed by:

Douglas L. Van Bossuyt  
Advisor

Ronald E. Giachetti  
Co-Advisor

Giovanna Oriti  
Co-Advisor

Daniel Reich  
Co-Advisor

Mark M. Rhoades  
Co-Advisor

Corina L. White  
Co-Advisor

Accepted by:

Oleg A. Yakimenko  
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

## ABSTRACT

Military installation microgrids need to be resilient to a variety of potential disruptions (storms, attacks, et cetera). Various metrics for assessing microgrid resilience have been described in literature, and multiple tools for simulating microgrid performance have been constructed; however, it is often left to system owners and maintainers to bring these efforts together to identify and realize effective, efficient improvement strategies. Military microgrid stakeholders have expressed a desire for an integrated, unified platform that provides these multiple capabilities in a coordinated fashion. In support of these endeavors, analysis methods developed by NPS and NAVFAC Expeditionary Warfare Center researchers for measuring microgrid resilience have been integrated into an existing web-based microgrid power flow simulation and distributed energy resource rightsizing software tool. This was achieved by the development of additional functions and methods within the existing software platform code base, and expansion of the application programming interface (API). These API additions enabled access to the new calculation and analysis capabilities, as well as increased control over power flow simulation parameters. These analytical and functional contributions were validated through a design of experiments, including comparison to independently generated data, and factorial analysis.



THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

<b>I.</b>	<b>INTRODUCTION.....</b>	<b>1</b>
<b>A.</b>	<b>BACKGROUND .....</b>	<b>2</b>
<b>B.</b>	<b>PROBLEM DEFINITION .....</b>	<b>3</b>
<b>C.</b>	<b>LITERATURE REVIEW, RESEARCH, AND QUESTIONS .....</b>	<b>4</b>
<b>D.</b>	<b>METHODOLOGY .....</b>	<b>5</b>
<b>E.</b>	<b>SCOPE AND DELIVERABLES .....</b>	<b>6</b>
<b>II.</b>	<b>SYSTEMS ENGINEERING AND ANALYSIS .....</b>	<b>7</b>
<b>A.</b>	<b>SYSTEMS ENGINEERING PROCESSES.....</b>	<b>7</b>
<b>1.</b>	<b>Waterfall .....</b>	<b>7</b>
<b>2.</b>	<b>Agile.....</b>	<b>8</b>
<b>3.</b>	<b>Team Systems Engineering Process Selection .....</b>	<b>9</b>
<b>B.</b>	<b>STAKEHOLDER ANALYSIS .....</b>	<b>11</b>
<b>C.</b>	<b>REQUIREMENTS IDENTIFICATION .....</b>	<b>12</b>
<b>D.</b>	<b>FUNCTIONAL ANALYSIS .....</b>	<b>16</b>
<b>E.</b>	<b>PROJECT VISION AND USE CASE DEVELOPMENT .....</b>	<b>20</b>
<b>1.</b>	<b>Vignette 1: Administrative User .....</b>	<b>21</b>
<b>2.</b>	<b>Vignette 2: Technical User .....</b>	<b>22</b>
<b>F.</b>	<b>USE CASES.....</b>	<b>23</b>
<b>1.</b>	<b>Calculate Microgrid Resilience (Use Case 6, MVP) .....</b>	<b>24</b>
<b>2.</b>	<b>Create Disturbance Scenarios (Use Case 2).....</b>	<b>27</b>
<b>3.</b>	<b>Create DER Models (Use Case 3) .....</b>	<b>27</b>
<b>4.</b>	<b>Calculate Microgrid Cost Estimate (Use Case 7).....</b>	<b>28</b>
<b>G.</b>	<b>SYSTEMS ENGINEERING MANAGEMENT PROCESSES .....</b>	<b>29</b>
<b>1.</b>	<b>Schedule .....</b>	<b>29</b>
<b>2.</b>	<b>Sprint Schedule Detail .....</b>	<b>31</b>
<b>3.</b>	<b>Risks .....</b>	<b>32</b>
<b>4.</b>	<b>Software Configuration Management.....</b>	<b>34</b>
<b>III.</b>	<b>SOFTWARE ARCHITECTURE AND DESIGN .....</b>	<b>39</b>
<b>A.</b>	<b>FOUNDATIONAL SOFTWARE.....</b>	<b>39</b>
<b>B.</b>	<b>RESILIENCE METRICS CALCULATION ADDITIONS .....</b>	<b>41</b>
<b>1.</b>	<b>Invulnerability.....</b>	<b>41</b>
<b>2.</b>	<b>Recoverability.....</b>	<b>47</b>
<b>3.</b>	<b>Resilience .....</b>	<b>49</b>
<b>4.</b>	<b>Supporting Functions .....</b>	<b>50</b>

C.	<b>APPLICATION PROGRAMMING INTERFACE AUGMENTATIONS .....</b>	<b>54</b>
IV.	<b>ANALYSIS .....</b>	<b>61</b>
A.	<b>MODEL VALIDATION .....</b>	<b>61</b>
B.	<b>EXPANDED CHARACTERIZATION .....</b>	<b>66</b>
C.	<b>CHARACTERIZATION RESPONSES .....</b>	<b>69</b>
V.	<b>CONCLUSIONS .....</b>	<b>77</b>
A.	<b>RESEARCH QUESTIONS .....</b>	<b>77</b>
1.	<b>Effective Integration of Tool Platform.....</b>	<b>77</b>
2.	<b>Tool Platform Accessibility .....</b>	<b>77</b>
3.	<b>Tool Platform Validity.....</b>	<b>78</b>
B.	<b>SOFTWARE DEVELOPMENT ARTIFACTS .....</b>	<b>78</b>
1.	<b>Upstream Repository Merging .....</b>	<b>79</b>
2.	<b>Software and Operating Environment Requirements.....</b>	<b>80</b>
3.	<b>Web Server Infrastructure.....</b>	<b>82</b>
C.	<b>HYBRID AGILE METHOD .....</b>	<b>83</b>
D.	<b>RECOMMENDATIONS FOR FUTURE WORK.....</b>	<b>83</b>
1.	<b>Technical Improvements to Resilience Modeling .....</b>	<b>84</b>
2.	<b>Technical Improvements to MPT Suite .....</b>	<b>85</b>
3.	<b>Deferred Stakeholder Needs .....</b>	<b>89</b>
4.	<b>Deferred System Requirements .....</b>	<b>89</b>
	<b>APPENDIX A. SYSTEM REQUIREMENTS.....</b>	<b>91</b>
	<b>APPENDIX B. STAKEHOLDER NEEDS TO SYSTEM REQUIREMENTS TRACEABILITY.....</b>	<b>97</b>
	<b>APPENDIX C. EXPANDED MPT API DOCUMENTATION .....</b>	<b>105</b>
	<b>APPENDIX D. EXAMPLE SCRUM DETAILS.....</b>	<b>125</b>
	<b>LIST OF REFERENCES.....</b>	<b>127</b>
	<b>INITIAL DISTRIBUTION LIST .....</b>	<b>131</b>

## LIST OF FIGURES

Figure 1.	Software Waterfall SE Process. Source: Buede (2009, 21).....	8
Figure 2.	High Level Agile Project Development Cycle. Source: Kukhnavets (n.d.).....	9
Figure 3.	Initial Team SE Process Approach .....	10
Figure 4.	Microgrid Resilience Analysis Software Development Performative Diagram. Adapted from Peterson et al. (2021) and SouthWest Water Company (2021). .....	15
Figure 5.	Top-Level IDEF0 Diagram.....	17
Figure 6.	Decomposed F.1 Simulate and Analyze Microgrid IDEF0 Diagram .....	18
Figure 7.	Overall Asset Diagram.....	19
Figure 8.	Decomposed M.1 MPT Asset Diagram .....	20
Figure 9.	Installation Manager Operational Vignette. Adapted from SouthWest Water Company (2021) and TONEX (n.d.).....	22
Figure 10.	Engineer Operational Vignette. Adapted from TONEX (n.d.) and Peterson et al. (2021). .....	23
Figure 11.	Microgrid Resilience Analysis Software Development Use Case Diagram.....	24
Figure 12.	UC6 Calculate Microgrid Resilience Action Diagram .....	26
Figure 13.	UC2 Create Disturbance Scenarios Action Diagram.....	27
Figure 14.	UC3 Create DER Models Action Diagram.....	28
Figure 15.	Project Schedule.....	30
Figure 16.	Risk Rating Matrix.....	32
Figure 17.	Fork of MPT Software Repository .....	34
Figure 18.	Software Commit Activity Excerpt .....	35
Figure 19.	Side-by-Side Comparison of a Commit in GitLab .....	36

Figure 20.	Microgrid Resilience Analyst Software Development Project Git Branches.....	37
Figure 21.	Microgrid Planning Tools Power Flow Simulation Web Interface. Source: Reich (n.d.) .....	40
Figure 22.	Ideal Power Flow Curve with Disturbance.....	42
Figure 23.	Simulated Power Flow Curve with Disturbance.....	43
Figure 24.	Invulnerability Calculation Python Code.....	44
Figure 25.	Input Variable Definitions for Invulnerability Calculation Python Code .....	45
Figure 26.	Minimum Power Delivered During Disturbance Impact Determination Python Code (1/2).....	45
Figure 27.	Minimum Power Delivered During Disturbance Impact Determination Python Code (2/2).....	46
Figure 28.	Input Variable Definitions for Minimum Power Delivered During Disturbance Impact Determination Python Code .....	46
Figure 29.	Recoverability Calculation Python Code.....	48
Figure 30.	Input Variable Definitions for Recoverability Calculation Python Code .....	49
Figure 31.	Resilience Calculation Python Code.....	49
Figure 32.	Input Variable Definitions for Resilience Calculation Python Code.....	50
Figure 33.	Disturbance Start and End Time Determination Conditional Structure Python Code .....	51
Figure 34.	Input Variable Definitions for Disturbance Start and End Time Determination Conditional Structure Python Code .....	51
Figure 35.	Disturbance Start Time Search Python Code.....	52
Figure 36.	Input Variable Definitions for Disturbance Start Time Search Python Code .....	52
Figure 37.	Single Simulation Run Resilience Metrics Value List Storage Python Code .....	53
Figure 38.	Resilience Metrics Values Statistical Calculations Python Code.....	53

Figure 39.	Input Variable Definitions for Single Simulation Run Resilience Metrics Value List Storage and Resilience Metrics Values Statistical Calculations Python Code.....	54
Figure 40.	API Example of Querying Pre-defined Microgrid Composition.....	57
Figure 41.	API Example of Querying Available Power Load Profiles .....	58
Figure 42.	API Example of Querying Available Microgrid Disturbances.....	58
Figure 43.	API Example of Executing a Microgrid Power Flow Simulation Series.....	60
Figure 44.	Design Method Validation. Source: Pedersen et al. (2000, 6).....	61
Figure 45.	MPT and Anderson’s Cost Model ANOVA.....	63
Figure 46.	Resilience Main Effects for Anderson Model .....	64
Figure 47.	Resilience Pareto Chart for Anderson Model .....	65
Figure 48.	Resilience Interaction Plot for Anderson Model .....	65
Figure 49.	DER Configuration Parameters .....	67
Figure 50.	Analyzed Power Load Profiles .....	69
Figure 51.	Resilience Metrics Main Effects.....	71
Figure 52.	Invulnerability Effects Pareto Chart .....	72
Figure 53.	Recoverability Effects Pareto Chart.....	73
Figure 54.	Resilience Effects Pareto Chart .....	73
Figure 55.	Resilience Metrics Effects Interaction Plot.....	75
Figure 56.	Upstream Merge Request with Original MPT Repository .....	80
Figure 57.	MPT Development Server Status Information Sample .....	82
Figure 58.	CPU Usage During Microgrid Simulation.....	87
Figure 59.	API Endpoint Interaction Example Jupyter Notebook .....	88
Figure 60.	Example Sprint Planning and Retrospective Board.....	125

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF TABLES

Table 1.	Stakeholder Identification.....	12
Table 2.	Stakeholder Needs .....	14
Table 3.	Sprint Schedule List and Goals.....	31
Table 4.	Risk Summary.....	33
Table 5.	MPT Resilience Metric Validation DOE Factors and Values .....	63
Table 6.	MPT Resilience Metric Characterization DOE Factors and Values.....	66
Table 7.	Disturbance Failure Rates .....	68
Table 8.	List of MPT Software Dependencies and Versions.....	81
Table 9.	System Requirements.....	91
Table 10.	Stakeholder Needs to System Requirements Traceability .....	97



THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF ACRONYMS AND ABBREVIATIONS

ANOVA	Analysis of Variance
API	Application Programming Interface
CNIC	Commander, Navy Installation Command
DASN(OE)	Deputy Assistant Secretary of the Navy (Operational Energy)
DER	Distributed Energy Resource
DOD	Department of Defense
EEDMI	Expected Electrical Disruption Mission Impact
ESF	Energy Security Framework
EXWC	Expeditionary Warfare Center
FPR	Final Progress Review
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
IPR	Interim Progress Review
JSON	JavaScript Object Notation
KPP	Key Performance Parameter
MBSE	Model Based Systems Engineering
MDI	Mission Dependency Index
MIL-STD	Military Standard
MPT	Microgrid Planning Tools
MSOSA	Magic Systems of Systems Architect
MTBF	Mean Time Between Failures
NAS	Naval Air Station
NAVFAC	Navy Facilities Engineering Systems Command
NPS	Naval Postgraduate School
NS	Naval Station
NSETTI	Navy Shore Energy Technology Transition and Integration Program

PMP	Project Management Plan
SE	Systems Engineering
SQL	Structured Query Language
TPO	Thesis Processing Office
UI	User Interface
URL	Uniform Resource Locator

## EXECUTIVE SUMMARY

Defense elements depend on reliable energy delivery to enable operational capability, deliver critical functions, and maintain services (Office of the Assistant Secretary of Defense for Energy, Installations and Environment 2015). The extent or capacity to which a relevant system can reliably meet critical demand is termed energy security (Naval Facilities Engineering Systems Command 2021; Siritoglou, Oriti, and Van Bossuyt 2021; Cullom 2009). Microgrids offer a unique solution to resist such disturbances (Peterson et al. 2021). Microgrids are integrated, self-contained power management systems consisting of energy storage and generation sources, supported loads, and a controller (Hirsch, Parag, and Guerrero 2018) that may or may not be connected to larger regional electrical grids. Microgrids may improve energy security through the modularity, diversification, and redundancy of electrical grid components.

Prior research has led to a better understanding of energy security (Siritoglou, Oriti, and Van Bossuyt 2021; Cullom 2009), the development of vulnerability mitigation strategies (Anglani et al. 2021; Reich and Oriti 2021; Kain, Van Bossuyt, and Pollman 2021), and improved descriptive techniques seeking to create a universal language and methods for analyzing microgrids (Peterson et al. 2021; Giachetti et al. 2022). These frameworks employ quantitative processes to calculate metrics of recoverability, invulnerability, and resilience. However, these tools all contain various usability and ease of access concerns. This work sought to provide these descriptive metrics in an easily accessible and unified platform, appropriate for microgrid planning efforts.

The Microgrid Planning Tool (MPT) Suite product provided a suitable foundation within which to add resilience, invulnerability, and recoverability functions. The suite's power flow simulation and statistical examination framework (Reich n.d.) provided a robust software base to build upon. The choice of intuitive software language (Python) and use of modern software development practices made it the ideal target for implementing these resilience analysis capabilities. As part of this project, updates were made to the simulation structure of the MPT suite to facilitate resilience calculations. A well-

documented API interface was also developed to inform user-level access to simulation configuration and execution, and resultant data retrieval.

The MPT suite provides a user-facing web front end, accessible from any web browser. The APIs assembled as part of this project provide the framework necessary for a web developer to enhance this interface with the added resilience calculation and analysis capabilities. While an updated web interface was not developed as part of this project, Jupyter notebooks (essentially Python code snippets) were constructed that allowed a user with moderate software experience to fully interact with the improved MPT suite. Whether via web UI or Jupyter notebook, the suite retains the server architecture of a web platform where centralized updates are available without user intervention, and data storage allows distribution and sharing of various microgrid profiles.

The justification process for validation of the MPT microgrid resilience calculation additions was driven by utility and usefulness, rather than formal accuracy. Structurally, the tool's internal logic has already withstood peer review and gained academic acceptance (Reich and Oriti 2021). However, performance evaluation proved more challenging. Because the tool evaluates microgrid performance under unlikely events, there is sparse real-world data available for comparison. Therefore, the tool was benchmarked against a similar Excel-based application, Dr. William Anderson's microgrid resilience and cost model (Anderson 2020). Based on configurations and scenarios supported by both tools, a partial factorial design of experiments (DOE) was assembled and executed. An analysis of variance (ANOVA) revealed that primary behaviors agreed between the two tools, though model type did produce a noted difference in the resilience response. Speculatively, certain simulated microgrid configurations may violate internal constraints or intended use cases of Anderson's model. Slight differences in the mathematical interpretation of ambiguities in the definitions for invulnerability, recoverability, and resilience also likely accounted for observed variations.

A more general characterization of the new MPT resilience calculation capability was obtained through full factorial analysis. Manipulated factors included energy resources (diesel generators, photovoltaic generators, and batteries), microgrid disturbance events, and microgrid power load profiles. Factors were isolated to determine their respective

independent effects on resilience metric responses. The first, and most significant, observation was a high sensitivity to incrementing from a single to a dual diesel generator set, resulting in a drastic increase to microgrid resilience. Second, a constant, moderate improvement to resilience was noted by continually increasing battery quantity. Third, increasing the photovoltaic generator quantity correlated only slightly positively with returned resilience values. Finally, disturbance types and power loads were associated with changes in resilience as expected. Disturbances featuring high probabilities of impact to energy resources, and power load profiles with higher average power demand, produced lower resilience metric responses, and vice versa.

Recommendations for future work include considerations of the details of the resilience metric calculations. As currently defined, invulnerability is represented by the ratio of singular values of power delivered and power demanded; however, both of these values are time-varying, and an accepted standard for reducing sets of these data to singular respective values has not yet been established. Given that disruption effects play a central role in the analysis of microgrid resilience, expanding resilience metrics to include probabilities for disturbance types, based on locality, would improve their fidelity, and would offer critical decision makers the ability to better quantify resulting mission impacts (alternative measures of resilience, such as the expected electrical disruption mission impact [EEDMI] (Peterson et al. 2021), already do this).

Additional suggestions for future efforts encompass further enhancements to the MPT suite. Incorporating the ability to define and simulate non-trivial microgrid architectures would enable more complex, and realistic, resilience analyses (currently, MPT only supports “single-node” architectures, wherein all energy resources are centrally connected). Related, the ability to specify load shedding strategies—establishing how and when non-critical power loads are to be disconnected from the rest of the microgrid when power generation capacity is not sufficient to meet overall power demand—would also enable improvements to MPT’s resilience analysis capability.

## References

- Anderson, William W. 2020. “Resilience Assessment of Islanded Renewable Energy Microgrids.” PhD diss, Naval Postgraduate School. <http://hdl.handle.net/10945/66574>.
- Anglani, Norma, Giovanna Oriti, Ruth Fish, and Douglas L. Van Bossuyt. 2021. “Design and Optimization Strategy to Size Resilient Stand-Alone Hybrid Microgrids in Various Climatic Conditions.” In *2021 IEEE Energy Conversion Congress and Exposition (ECCE)*, 210–17. Vancouver, BC, Canada: IEEE. <https://doi.org/10.1109/ECCE47101.2021.9595049>.
- Cullom, RADM Philip H. 2009. “Energy Security: Strategic Overview.” Slides, October 14, 2009. <https://www.nre.navy.mil/media/document/radm-cullom-energy-security-overviewpdf>.
- Giachetti, Ronald E., Douglas L. Van Bossuyt, William Anderson, and Giovanna Oriti. 2022. “Resilience and Cost Trade Space for Microgrids on Islands.” *IEEE Systems Journal* 16 (3): 3939–49. <https://doi.org/10.1109/JSYST.2021.3103831>.
- Hirsch, Adam, Yael Parag, and Josep Guerrero. 2018. “Microgrids: A Review of Technologies, Key Drivers, and Outstanding Issues.” *Renewable and Sustainable Energy Reviews* 90 (March): 402–11. <https://doi.org/10.1016/j.rser.2018.03.040>.
- Kain, Alissa, Douglas L. Van Bossuyt, and Anthony Pollman. 2021. “Investigation of Nanogrids for Improved Navy Installation Energy Resilience.” *Applied Sciences* 11 (9): 1–30. <https://doi.org/10.3390/app11094298>.
- Naval Facilities Engineering Systems Command. 2021. “3 Pillars of Energy Security (Reliability, Resilience, & Efficiency).” Washington Navy Yard, DC: Naval Facilities Engineering Systems Command. <https://www.wbdg.org/ffc/navy-navfac/p-publications/p-602>.
- Office of the Assistance Secretary of Defense for Energy, Installations and Environment. 2015. *2016 DOD Operational Energy Strategy*. Washington, DC: Office of the Assistance Secretary of Defense for Energy, Installations and Environment. <https://www.acq.osd.mil/eie/Downloads/OE/2016%20DoD%20Operational%20Energy%20Strategy%20WEBc.pdf>.
- Peterson, Christopher J., Douglas L. Van Bossuyt, Ronald E. Giachetti, and Giovanna Oriti. 2021. “Analyzing Mission Impact of Military Installations Microgrid for Resilience.” *Systems* 9 (3): 1–19. <https://doi.org/10.3390/systems9030069>.
- Reich, Daniel. n.d. “A Suite of Tools.” Migrogrid Planning Tools. Accessed October 8, 2022. <https://microgrid.nsetti.nps.edu/>.

Reich, Daniel, and Giovanna Oriti. 2021. "Rightsizing the Design of a Hybrid Microgrid." *Energies* 14 (14): 4273. <https://doi.org/10.3390/en14144273>.

Siritoglou, Petros, Giovanna Oriti, and Douglas L. Van Bossuyt. 2021. "Distributed Energy-Resource Design Method to Improve Energy Security in Critical Facilities." *Energies* 14 (May): 2773. <https://doi.org/10.3390/en14102773>.



THIS PAGE INTENTIONALLY LEFT BLANK

## ACKNOWLEDGMENTS

The team is extremely grateful to Dr. Oleg A. Yakimenko and Dr. Douglas Van Bossuyt for their advisement and guidance in developing this project. Additionally, we could not have accomplished this project without Dr. Daniel Reich's technical foundation to build upon. We are incredibly thankful to Dr. Reich for sharing his time and technical expertise with us, and for allowing us to contribute to his project. We would also like to thank Dr. Ronald Giachetti, Dr. Giovanna Oriti, Dr. William Anderson, Prof. Corina White, Rabia Kahn, and especially Prof. Mark Rhoades for their proofreading, insights, technical inputs, and assistance. Without their help, this would have been a lesser paper.

Tom would like to thank his wife, Kenzie, for her unwavering support both during this capstone project and years of classes. Her encouragement got him through many late-night coding sessions. And to his young boys, Aiden and Nathaniel, for giving up their Saturday mornings and family time so "Dad could go to school."

Dave would like to thank his partner, Nancy, for her constant encouragement and continued tolerance of frozen pizza dinners; his "bonus son," Ronin, for his enlightening curiosity and understanding of video game sessions cut short; and his outstanding capstone team, for their dedicated efforts and patience with his many neuroses. Dave would also like to send gratitude to Matthew Bedel, April White, and Stephen Pelletier, for their support and accommodation of his academic goals; and Dr. Matthew Gadlage, Dr. Adam Duncan, Dr. Matthew Kay, and Dr. Austin Roach, for being exceptional mentors and role models.

Mike would like to acknowledge the patience and grace provided by family and friends. Additionally, without the flexibility and support of the Maritime Electromagnetic Warfare Command, continued academic pursuits would not be possible.

Andrew would like to thank his wife, Lauren, for her continuous support and endless patience throughout the years of this program.

Rick would like to thank Dr. Douglas Van Bossuyt and Prof. Corina White for their guidance and support throughout the capstone process. Also a huge thank you to fellow teammates for understanding and working through busy schedules and travel requirements.

THIS PAGE INTENTIONALLY LEFT BLANK

## I. INTRODUCTION

Department of Defense (DOD) combatant and support elements, domestic or abroad, permanent or forward operating, rely on energy for training, movement, and sustainment of military forces and weapon platforms. The physical and logical infrastructure required to support energy demand is highly interconnected and complex. Disruptions occurring at a local element can potentially produce widespread, cascading effects, depending on timing, criticality, and resource buffering (Cornell 2009). Energy reliance, volatility, and an increasing concern of external, targeted disturbances led to the enactment of DOD-wide guidance to ensure mission readiness through the pursuit of energy security and energy resilience. Microgrids offer a unique solution to improve energy posture through the diversification of energy sources, grid redundancy, and by providing the modularity required to support rapid deployment. Microgrids are integrated energy systems consisting of interconnected loads and energy sources that can function independently of a traditional electrical utility grid. Efforts to optimize microgrid design have led to the development of various tools that inform the end user of performance sensitivity to component and architectural changes. These techniques, along with an adequately advanced control system, can be used to ensure critical equipment is supported when the entire load cannot be maintained, thus bolstering the likelihood of continued effective capability.

Currently, these tools are provided in a stand-alone fashion. The installation manager, engineer, or other vested stakeholder is responsible for identifying and understanding each tool, generating and analyzing the metrics of importance, and selecting a design that meets their specific use case and performance requirements. To this end, the intent of the work presented in this report is to offer a unified experience to the end user, enhancing the functionality of an existing Microgrid Planning Tool (MPT) software platform through the addition and integration of a novel resilience calculation capability.

## A. BACKGROUND

Energy security is characterized as reliable access to energy delivery, through generation or storage, in order to fulfill mission critical functions and services (Naval Facilities Engineering Systems Command 2021; Office of the Assistant Secretary of Defense for Energy, Installations and Environment 2015; Siritoglou, Oriti, and Van Bossuyt 2021; Cornell 2009; Cullom 2009). Fundamentally, energy security is dependent on the confluence of three determining factors: reliability, efficiency, and resilience. In accordance with Energy Security Framework (ESF) implementation, Naval Facilities Engineering Systems Command (NAVFAC) sought to establish a consistent process for installation energy evaluation. Their process framed the aforementioned factors as “pillars,” utilized to establish standardized metrics and assessment techniques. In the interest of clarity, definitions are provided as follows. Reliability is described as “the percentage of time energy delivery systems (utilities) can serve mission functions at acceptable mission essential requirements, as well as serve customers at acceptable regulatory standards,” and efficiency as “use of the minimal energy required to achieve the desired level of service” (Naval Facilities Engineering Systems Command 2021, 42, 44). The definition offered for resilience, borrowed from 10 U.S.C. § 101(e)(7), is “the ability to avoid, prepare for, minimize, adapt to, and recover from anticipated and unanticipated energy disruptions in order to ensure energy availability and reliability sufficient to provide for mission assurance and readiness, including mission essential operations related to readiness, and to execute or rapidly reestablish mission essential requirements” (Naval Facilities Engineering Systems Command 2021, 43). A more succinct definition for resilience is provided in DOD Instruction 4170.11, as “the ability to prepare for and recover from energy disruptions that impact mission assurance on military installations” (Department of Defense 2009, 24). A primary tactic for improving the energy resilience, and thus the energy security, of the DOD’s operational establishments is the implementation of microgrids (Peterson et al. 2021), which are essentially reduced-size, self-contained electrical utility grids capable of being disconnected and operating independently from larger regional-level utility grids, when enacting a procedure known as “islanding” (Hirsch, Parag, and Guerrero 2018). Energy resilience is achieved by

incorporating local grid architecture elements, including energy production, storage, distribution, and control mechanisms (Kain, Van Bossuyt, and Pollman 2021).

In continuing efforts to enhance microgrid energy resilience, the Naval Postgraduate School (NPS) in Monterey, CA, and the Naval Facilities Engineering and Expeditionary Warfare Center (NAVFAC EXWC) in Port Hueneme, CA, have developed methods and tools for analyzing and offering improvements to the resilience of existing and in-planning microgrid systems. The work presented in this report focuses on augmenting these tools with metrics developed for measuring the resilience of given microgrid architectures, offering an integrated environment within which to perform microgrid improvement analyses. Primarily, this involved iterating upon previous work sponsored by the NAVFAC EXWC Navy Shore Energy Technology Transition and Integration (NSETTI) program. Prior developments produced a web-based modeling and simulation tool (titled the “Microgrid Planning Tool,” or MPT (Reich n.d.)), which generates optimal distributed energy resource (DER) allocations. These calculations are based on user selected DER performance specifications and historical, location-based power load profiles. We leveraged and improved upon existing logic, functions, and capabilities, in order to add and incorporate resilience determination and analysis capacity.

## **B. PROBLEM DEFINITION**

Military installation microgrids need to be resilient to a variety of potential electrical power disruptions, such as natural causes as well as adversary action (Office of the Assistant Secretary of Defense for Energy, Installations and Environment 2015) and are a key planning and management focus for installation commanders and facilities support administrators (e.g., NAVFAC). Research to date has developed many stand-alone modeling and analysis capabilities to look at various disruptive impacts and supply chain constraints for electrical power. These methods and tools can be used to compare the resilience of differing microgrid architectures.

However, these approaches and resources are not available in an easy-to-use fashion for installation planners. Typically, they are tied to a targeted study with little effort to provide decision makers a flexible and easy-to-use tool. Additionally, the Navy desires

any tools to have a sound Model Based Systems Engineering (MBSE) foundation (Bickford et al. 2020), using the latest digital engineering techniques (United States Navy and Marine Corps 2020) to facilitate modeling, simulation, maintainability, and sustainability of the model. Toward this end, a systems engineering (SE) approach was developed and implemented to provide an integration capability for unifying these existent methods and approaches into a convenient and effective singularized flow. This work incorporates non-specialized review and planning tool sets, utilizing freely available standards to allow decision makers to interact with the SE process at their level, and builds on existing work by NPS, NAVFAC EXWC, DASN(OE), and others.

### **C. LITERATURE REVIEW, RESEARCH, AND QUESTIONS**

The concept of electrical microgrids, and their benefits to energy reliability and resilience, are not unique to military operations, and are presented in general form and practice by Hirsch, Parag, and Guerrero (2018). Peterson, Van Bossuyt, Giachetti, and Oriti (2021) develop a method specifically for quantifying military microgrid resilience in the context of national security mission impact, though it incorporates admitted subjectivity in the determination of load criticality, and pointedly avoids including cost considerations. An alternative resilience metric, incorporating measures of microgrid invulnerability and recoverability, is detailed by Giachetti, Van Bossuyt, Anderson, and Oriti (2022). The unique systems engineering challenges faced by military microgrid engineers and designers, and motivation for the holistic consideration thereof, are discussed by Giachetti, Van Bossuyt, Parker, and Peterson (2020).

Other related works have pursued further examination of microgrid resilience from multiple perspectives. Anuat, Van Bossuyt, and Pollman (2021) establish a metric for assessing the impact of a given energy resilience measure upon a specific critical load, and further, an entire microgrid. The effects of unique geographic climate conditions upon microgrids are investigated by Anglani, Oriti, Fish, and Van Bossuyt (2021), and a resultant strategy is provided for maximizing resilience through optimizing constituent DER allocation. Another such allocation strategy is offered by Reich and Oriti (2021), based on DER performance parameters. Nanogrids—effectively a lower-level extension of

microgrids, providing an even higher degree of localized DER capability, and consequently even greater resilience—are explored by Kain, Van Bossuyt, and Pollman (2021). Microgrid resilience cost implications have also been investigated, in the form of a cost tradeoff analysis (Giachetti et al. 2022), in terms of typical costs associated with providing/ensuring resilience (Hildebrand 2020), and from the standpoint of evaluating impacts to resilience from varying equipment costs (Bolen et al. 2021).

Of note, the above referenced methods, strategies, and approaches are largely disparate. While beneficial and worthwhile each in their own right, maximal value would be derived from a simultaneous implementation of all these microgrid resilience enhancing techniques. In light of these considerations, we propose the following list of research questions, to be addressed by this project:

1. What is an effective way to integrate these microgrid resilience analysis methods and tools into a unified, efficient, and useful platform?
2. How can intuitive, practical accessibility be provided for such a tool platform?
3. What measurements can be used to assess the validity of such a tool platform?

#### **D. METHODOLOGY**

Substantive integration related to research and models was required to develop a more ontologically complete microgrid planning platform. Foundational claims included the following. First, the existing MPT architecture is understood to agree with a subset of observable, single node microgrids. Second, evaluation techniques established in preceding research created a unified approach appropriate for generalized microgrid description.

Conceptualizations for resilience, recoverability, and invulnerability were tailored for, and constrained by, the MPT suite. Detailed interface definition and modification allowed shared resource exchange between the existing architecture and the resilience analysis additions. Unique functions required for resilience implementation were then assigned to developmental sprints. This process allowed for frequent, quantitative



assessment of isolated functions. If localized technical merit was confirmed, changes were propagated into the main software branch.

After complete model realization, validation was pursued to determine usefulness and utility. The selected method, use of the Validation Square (Pedersen et al. 2000), was a mixed framework consisting of qualitative and quantitative processes. Factorial analysis was used to characterize microgrid response sensitivity and provided the basis for platform comparisons.

## **E. SCOPE AND DELIVERABLES**

System life cycle generally encompasses research, development, production, deployment, support, and disposal phases, but due to this project's accelerated timeline it included only up to production. The team initiated the project with stakeholder involvement and research of prior work in order to progress efficiently through the project phases and meet targeted objectives. All software products were stored on NPS's GitLab data repository for controlled and assured access, reliable archival, and future retrieval. Supportability and/or updates will not be provided beyond the delivery of the validated end product. Code is well defined and annotated as to allow for future updates to be made. A capstone report (this document) has been made available to describe the work done, and to detail future work that could be performed to further advance the products.

## II. SYSTEMS ENGINEERING AND ANALYSIS

### A. SYSTEMS ENGINEERING PROCESSES

SE processes are typically generalized frameworks that are expected to be scoped to fit a project's needs and ensure that SE principles are followed during a project's life cycle. However, there are typically several process paradigms that can be selected from in order to ensure a project is completed in a timely manner, with a functional deliverable, and while minimizing risk.

The team explored two SE process approaches for software development, the waterfall approach and the Agile approach. The waterfall approach is a traditional SE process that is well understood, but can lack flexibility. The Agile approach is a more modern process that can increase flexibility at the expense of a well-structured work breakdown structure.

#### 1. Waterfall

The waterfall model is a linear approach to software design. It is an old approach, developed in the 1970s, and forms the basis of MIL-STD-2167A (Buede 2009). Buede goes on to say that a typical problem of waterfall designs is that the steps that are iterated upon can end up far apart from each other. This can manifest either in an extended effort timeline, or loss of work organization and coordination—the result is the same issue. A lack of communication between steps can lead to poorly designed systems that end up well behind schedule with little to show for effort. Figure 1 is a pictorial representation of this model. Not shown is that ideally these steps are happening in parallel as well, with requirements development for the next feature or release beginning once the previous one is further down the waterfall.

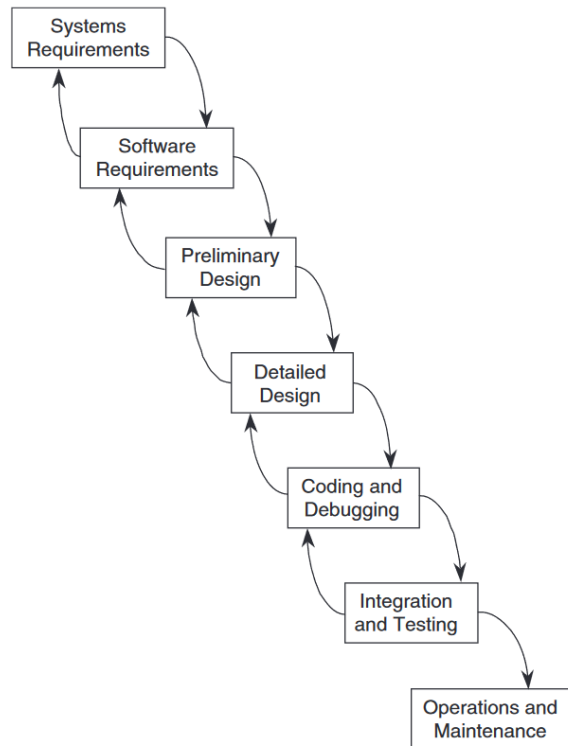


Figure 1. Software Waterfall SE Process. Source: Buede (2009, 21).

## 2. Agile

Agile software development is a newer approach that focuses on delivering incremental, functional software in regular intervals (Agile Alliance n.d.). Agile has additional benefits too. It ensures that the team does not take on more work than they are capable of, and requires that everyone maintain focus and communicate on tasking. It also allows teams to adapt to changing requirements and user needs. (Atlassian n.d.)

Figure 2 shows a typical high-level Agile approach to doing work. An initial project vision feeds the release plan. Sprints—short-term concise objective-focused execution cycles—are then run sequentially with their own contained planning, implementation, review, and retrospection. The output from each sprint should be a deployable or consumable item for the end user.

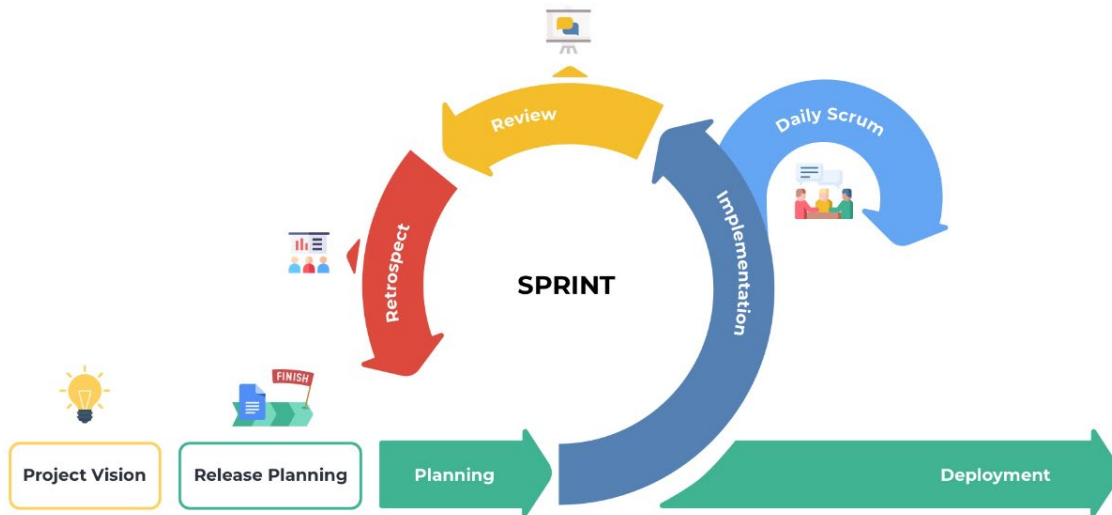


Figure 2. High Level Agile Project Development Cycle. Source: Kukhnavets (n.d.).

### 3. Team Systems Engineering Process Selection

Generally, academic projects are severely time constrained and not well defined beyond a project prompt. Additionally, it is easy to suffer scope creep as multiple stakeholders with very different requirements must be considered.

At first, the team attempted to use rapid iterations with an Agile-like sprint cycle to iterate on the scope of the project while incorporating stakeholder feedback in real-time. This early focus on requirements and initial design is similar to the first steps of a traditional waterfall SE process. Agile is also key to a Navy strategy to modernize systems engineering and delivery of capabilities to the warfighter (United States Navy and Marine Corps 2020). Figure 3 shows the team’s initial approach to developing an SE process to meet these constraints.

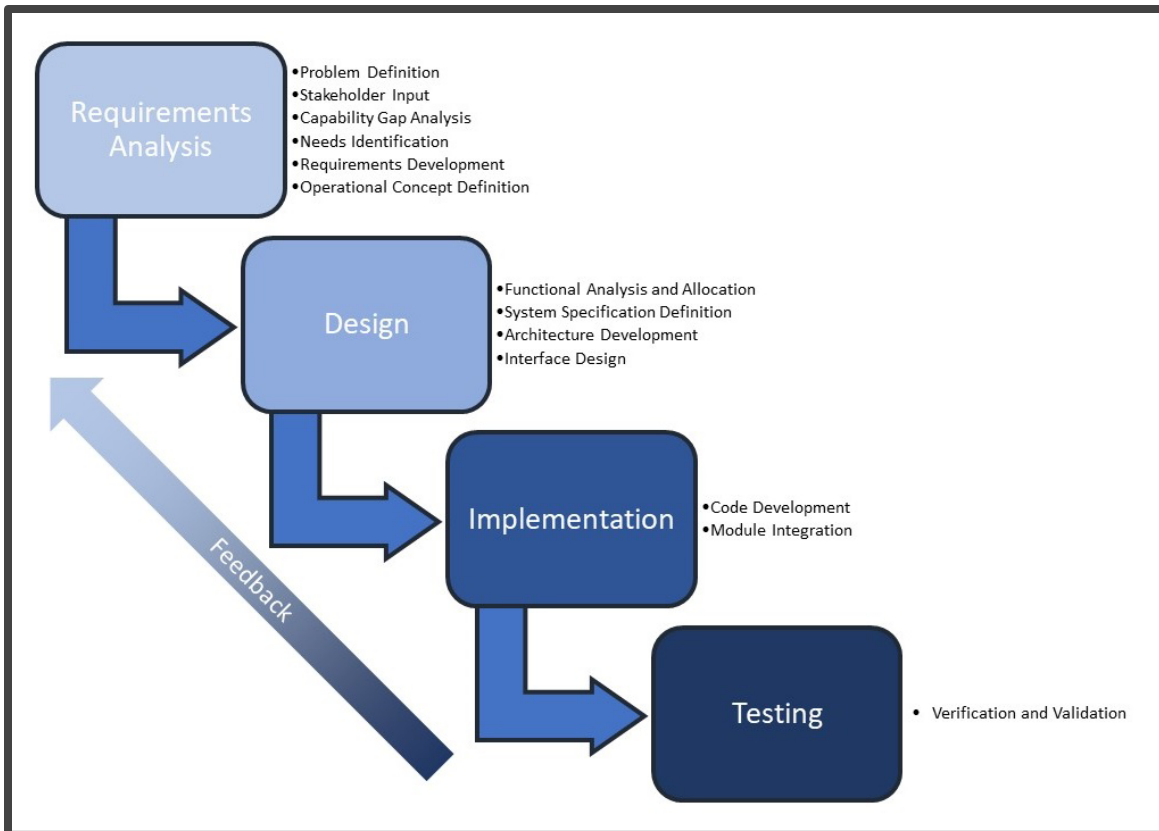


Figure 3. Initial Team SE Process Approach

However, this combined approach, maintaining aspects of a more traditional SE process, was determined to not be very agile. Within the first six weeks of the project, this approach began to show flaws. The large efforts in producing MBSE artifacts and keeping them up to date as the project scope was repeatedly modified based on stakeholder and advisor feedback became too much of a burden, and a reset to a proper Agile approach was recommended by the team’s advisors.

The team’s reorientation to a more “genuine” Agile approach, properly sized for the available resources and schedule, provided a much clearer idea of what the end state deliverables for the Microgrid Resilience Analysis Software Development project would look like, and received rapid buy-in from advisors and stakeholders.

## **B. STAKEHOLDER ANALYSIS**

In order to begin appropriately scoping the project, an understanding of the stakeholders involved, including their roles and respective requirements, was rapidly established. This early effort was handled in the traditional SE requirements analysis process. Using the project prompt as a problem definition, stakeholder analysis was conducted.

Stakeholders are defined as individuals, organizations, and technical groups that have significant interest in all aspects of the program (Blanchard and Fabrycky 2011; Buede 2009). Critically, improper stakeholder identification, or insufficient interaction, can lead to poor problem realization, derailing the success of the project. Therefore, the following process was developed to identify, communicate, analyze, and deliver stakeholder inputs.

1. Identify appropriate personnel within organizations that qualify as stakeholders.
2. Prioritize stakeholders by level of interest.
3. Ensure sponsors are prioritized correctly.
4. Collect inputs from stakeholders via needs and problem statements.
5. Ask identified stakeholders “What do you need?” and “What is the problem?”
6. Identify the scope boundaries.
7. Analyze and consolidate stakeholder input into actionable material.
8. Identify traceability to each of the stakeholder’s input.
9. Build the list of core needs the program should address.
10. Develop program requirements from the list of core needs.
11. Establish driving program needs to address the core stakeholder needs.

12. Consolidate program requirements, and ensure they are clear, concise, and traceable.
13. Communicate analyzed stakeholder inputs and requirements with stakeholders.
14. Incorporate feedback from stakeholders.
15. Repeat communication and feedback incorporation as necessary until all feedback is addressed appropriately.
16. Deliver final list of program requirements.

Stakeholders for this project were categorized as academic organizations, naval organizations, sponsors, or end users. Furthermore, stakeholders were tiered as either central or boundary members depending on level of interest, level of engagement, and programmatic influence. Table 1 identifies stakeholders and their respective relationships with the project.

Table 1. Stakeholder Identification

<u>Stakeholder</u>	<u>Academic</u>	<u>Navy</u>	<u>Sponsor</u>	<u>End User</u>	<u>Central</u>	<u>Boundary</u>
NPS	X	X			X	
NAVFAC EXWC		X				X
DASN(OE)		X	X		X	
CNIC		X				X
NS Rota		X		X		X
NAS Sigonella		X		X		X
NSETTI		X	X	X	X	

### C. REQUIREMENTS IDENTIFICATION

After stakeholder identification, an outreach message was distributed to stakeholders to inform project status, schedule individual meetings, and gather feedback.

The stakeholder needs are cornerstone requirements, that partition the problem into operational statements (Buede 2009), allowing for easier parallelization of work and providing the first level of decomposition. Table 2 lists stakeholder needs collected and recorded from participating members (the “Labels” column is explained near the end of this section, with the description of system requirements development and analysis).



Table 2. Stakeholder Needs

<u>Number</u>	<u>Name</u>	<u>Labels</u>
SN1	Stakeholder Needs	
SN1.1	Primary	
SN1.1.1	Improve DOD energy security analysis techniques.	Select (Threshold)
SN1.1.2	Evaluate resilience, recoverability, and invulnerability of microgrids.	Select (Threshold)
SN1.1.3	Advance toolkit to aid decision makers in microgrid design architecture.	Select (Threshold)
SN1.1.4	Provide interface tool to integrate digital model and web tool.	Select (Threshold)
SN1.2	Digital Model	
SN1.2.1	Create state machine for digital model control module.	Defer
SN1.2.2	Develop digital model system user manual for future efforts.	Defer
SN1.2.3	Update battery charge and discharge model.	Defer
SN1.2.4	Create dynamic critical and non-critical load function into digital model.	Defer
SN1.2.5	Create location-based aero function into digital model.	Defer
SN1.2.6	Create location-based irradiance function into digital model.	Defer
SN1.2.7	Create location-based climate function into digital model.	Defer
SN1.2.8	Create location-based power consumption function into digital model.	Defer
SN1.2.9	Calculate resilience metrics within digital model.	Defer
SN1.2.10	Update component and DER models.	Defer
SN1.2.11	Perform model validation and verification against similar digital models and experimental data.	Defer
SN1.2.12	Provide disturbance module within digital model.	Defer
SN1.3	Microgrid Planning Tool Module	
SN1.3.1	Create secure individual user profiles within web tool.	Defer
SN1.3.2	Create custom power profiles within web tool.	Defer
SN1.3.3	Create predefined power profiles within web tool.	Select (Threshold)
SN1.3.4	Create custom microgrid components within web tool.	Defer
SN1.3.5	Integrate SQL data input within web tool.	Defer
SN1.3.6	Simulate system disturbances within web tool.	Select (Threshold)
SN1.3.7	Calculate resilience metrics within web tool.	Select (Threshold)
SN1.3.8	Provide load shedding capability.	Defer
SN1.3.9	Update component and DER library within web tool.	Defer

Informational artifacts and the developed need statements form the basis of the operational concept, represented as a performative diagram. Shown in Figure 4, this graphically depicts the progression of microgrid resilience analysis, highlighting the sourcing, required element breakout, and processing flow of constituent data. Additionally, this diagram provides a means to orient and focus detailed discussions between stakeholders and the project team (Chief Information Officer, U.S. Department of Defense n.d.), ensuring a shared vision.

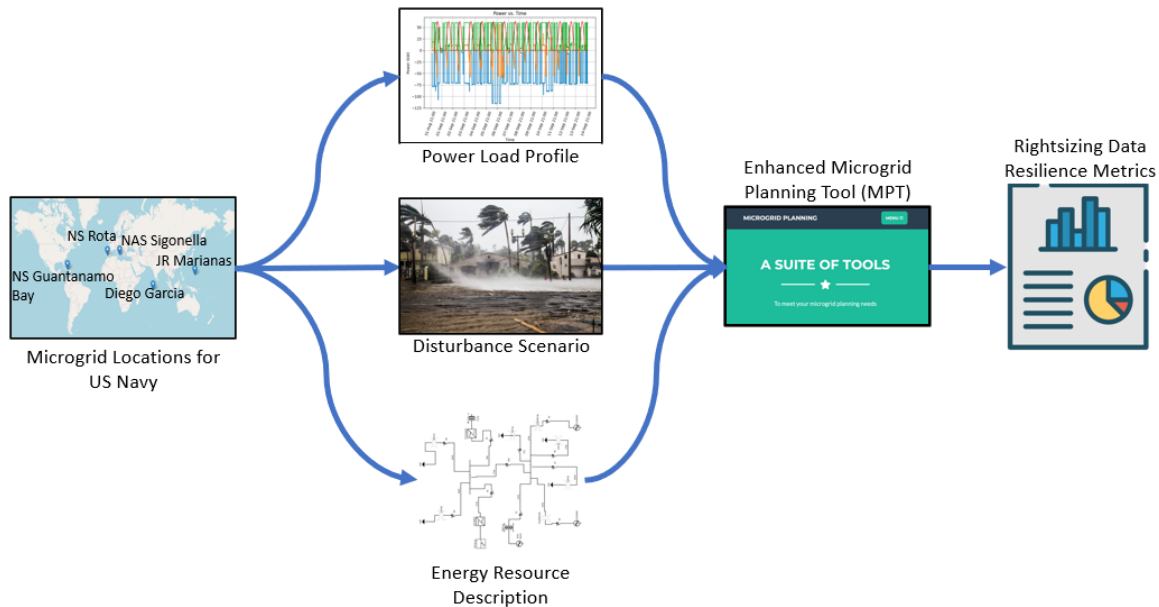


Figure 4. Microgrid Resilience Analysis Software Development Performative Diagram. Adapted from Peterson et al. (2021) and SouthWest Water Company (2021).

System views, requirements management, traceability, and functional allocation were handled by the Innoslate software suite (Innoslate n.d.). Innoslate allows for the creation of custom labels within the schema editor, which can be used to organize and manage SE process artifacts. This utility was leveraged to categorize stakeholder needs and system requirements. Categories consisted of “threshold” (will meet), “objective” (may meet, if time and capability allow), and “defer” (do not intend to meet). Deferred requirements were still tracked to build consensus, capture historical context, inform future

work, and identify capability gaps during functional allocation. The needs and operational concept were transformed through further decomposition, inductive reasoning, and iteration to generate system requirements. System requirements are provided in Table 9 in Appendix A, and a traceability mapping between stakeholder needs and system requirements is provided in Table 10 in Appendix B.

#### **D. FUNCTIONAL ANALYSIS**

Initially, functions were identified using stakeholder inputs and review of previous capstone material to show areas where current functionality exists and areas where development is required to meet overarching project goals. After development and refinement of stakeholder needs and system requirements, the initial set of functions were developed and incorporated into an IDEF0 diagram using Innoslate. These functions were separated into two primary functions.

1. Simulate and Analyze Microgrid
2. Construct Microgrid Systems Engineering Model

Each of these top-level functions were then decomposed into lower-level functions describing both the functional flow of the system and identifying required inputs and outputs. The top-level IDEF0 diagram is shown in Figure 5.

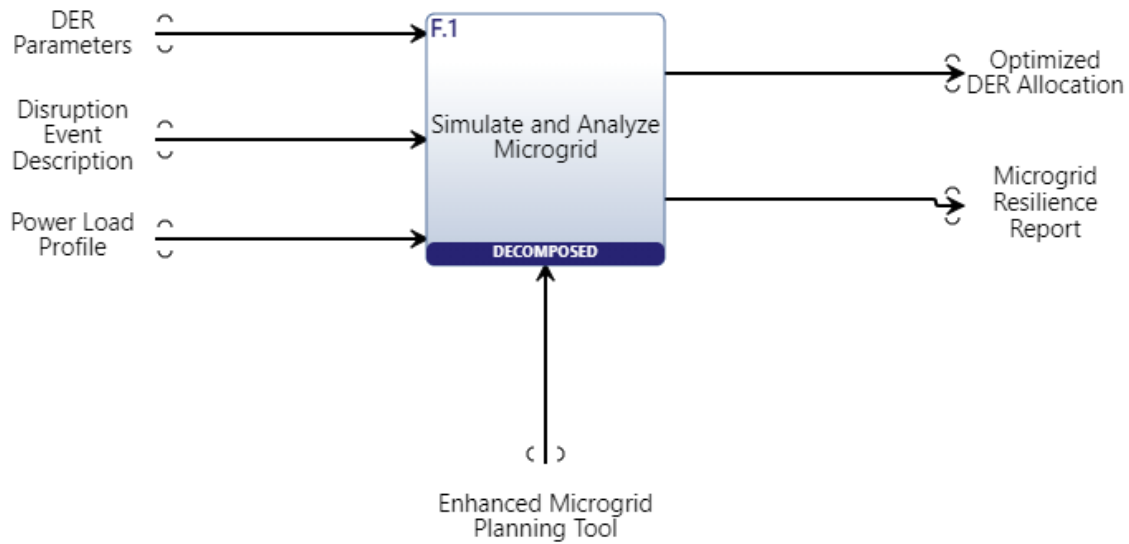


Figure 5. Top-Level IDEF0 Diagram

The top-level IDEF0 shows the primary inputs and outputs. Inputs include compositional DER parameters, microgrid power load profiles, and disruption events. Outputs consist of microgrid resilience reports and optimized DER allocation. Within the F.1 Simulate and Analyze Microgrid block, functions were decomposed to include parse user-entered data, simulate microgrid power generation and load, calculate optimized DER allocation, and calculate resilience metrics, as shown in Figure 6.

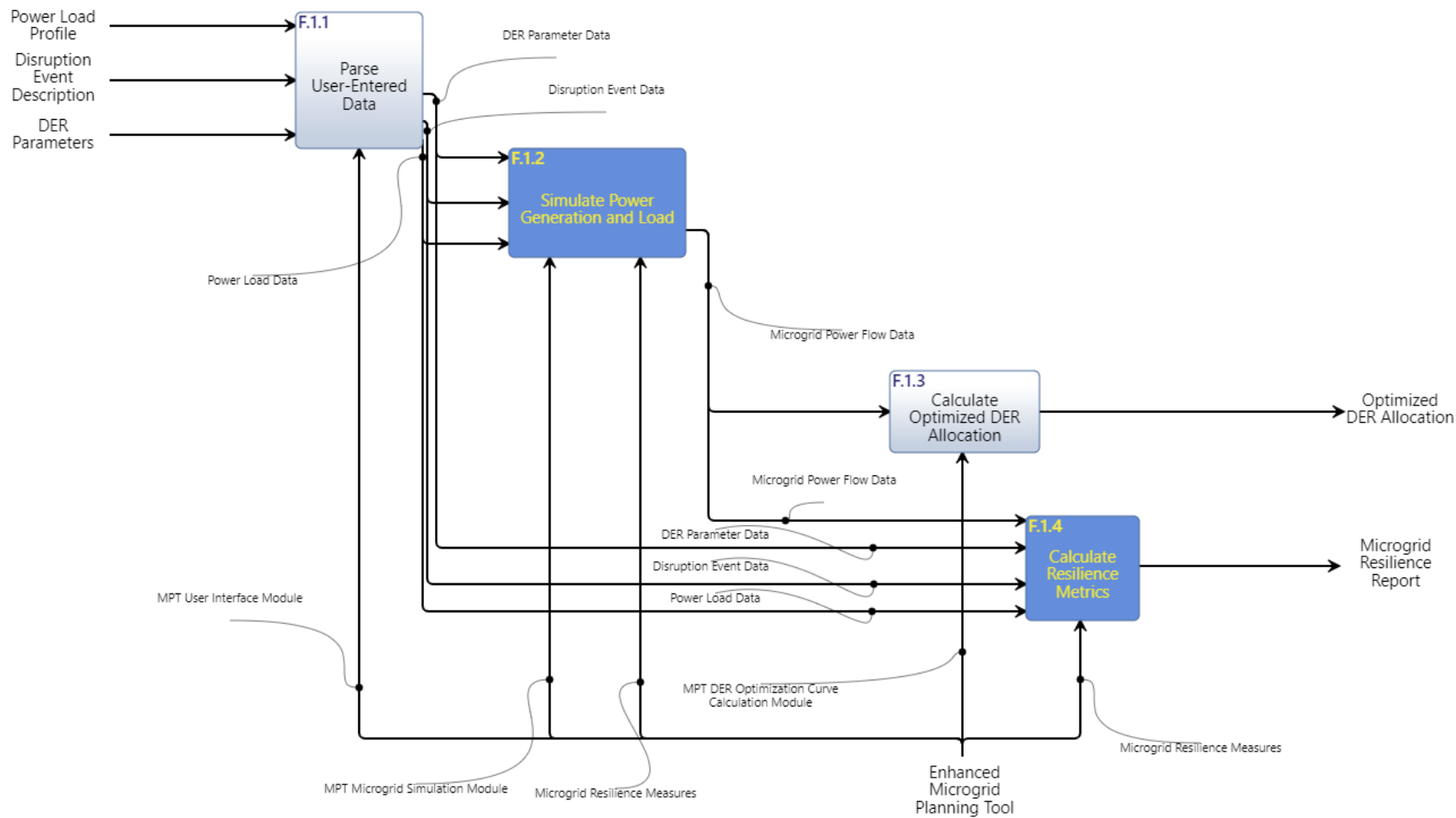


Figure 6. Decomposed F.1 Simulate and Analyze Microgrid IDEF0 Diagram

Various inputs and outputs, which are traced from the IDEF0, are shown within the asset diagram. Within the asset diagram, the source and destination for these inputs and outputs are shown, including system-external assets. The primary enhanced MPT asset was also decomposed into a lower-level diagram, as shown in Figure 7 and Figure 8.

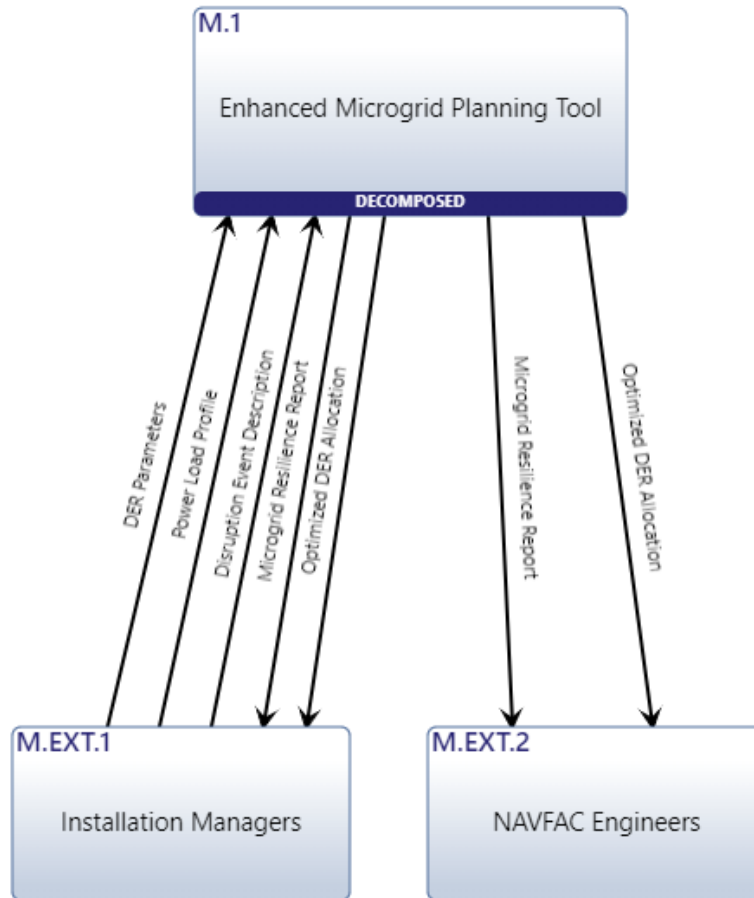


Figure 7. Overall Asset Diagram

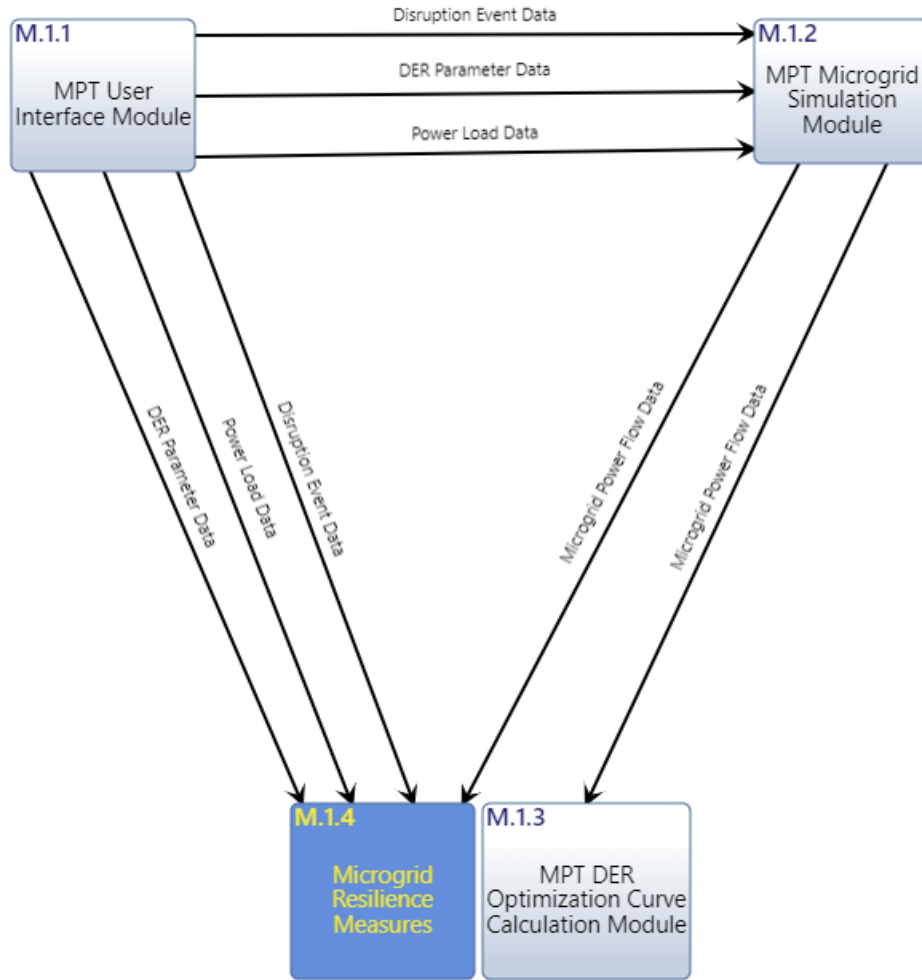


Figure 8. Decomposed M.1 MPT Asset Diagram

For the decomposition of the Microgrid Planning Tool, the user interface module interacts with the microgrid simulation model and the analysis and improvements methods module. Separately, the microgrid simulation model interacts with the DER optimization curve calculation model.

### E. PROJECT VISION AND USE CASE DEVELOPMENT

The cornerstone of an Agile software project is the project vision. For the MPT resilience metric calculation augmentations, this project’s vision was the OV-1 diagram shown in the Requirements Identification section (Figure 4). A project vision is interpreted

in much the same way as an OV-1 concept of operations diagram. It is an abstracted, non-technical vision of how a system will operate.

To further develop a non-technical description of the functionality of the targeted MPT enhancements, operational vignettes considering the end users of the system and outputs they might desire were constructed. These vignettes tell a story of how a user might see the capabilities of a system, still abstracted from a concordant MBSE view.

The team selected two vignettes to develop. One for a higher-level management user, who might advise decision makers; and one for an engineer, involved in the technical design and operation of a microgrid.

### **1. Vignette 1: Administrative User**

Figure 9 provides an operational vignette for a microgrid “installation manager.” Here the manager might be expected to play “what-if” scenarios using data already in the tool to answer data calls. Tweaks to the microgrid model could be used to advise decision makers on the pros and cons of available options to increase resilience or decrease costs. A key component of this is that the manager will not have access to low level engineering tools to do this analysis. A web browser or simple software package should suffice.



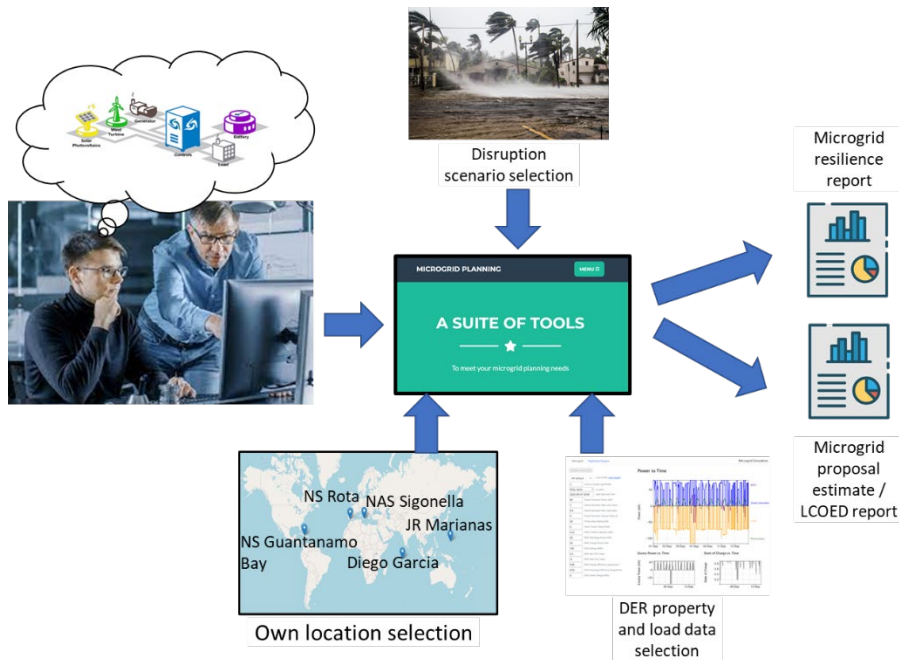


Figure 9. Installation Manager Operational Vignette. Adapted from SouthWest Water Company (2021) and TONEX (n.d.).

## 2. Vignette 2: Technical User

Figure 10 provides a lower-level operator vignette of MPT, such as would be used by a NAVFAC engineer or contractor design agent. Here, the engineer has knowledge of a digital system model of the microgrid, including its architecture and as-built layout. The engineer would also have access to microgrid DER load and power generation profiles, including generator capabilities, energy storage, and local solar irradiance values. The engineer would use the tool to collect direct resilience metrics to validate their model of a physical microgrid when disturbances occur or exercises are conducted. They would likely utilize Python or some other programming language in a simplified software development environment to work with the data they have obtained.

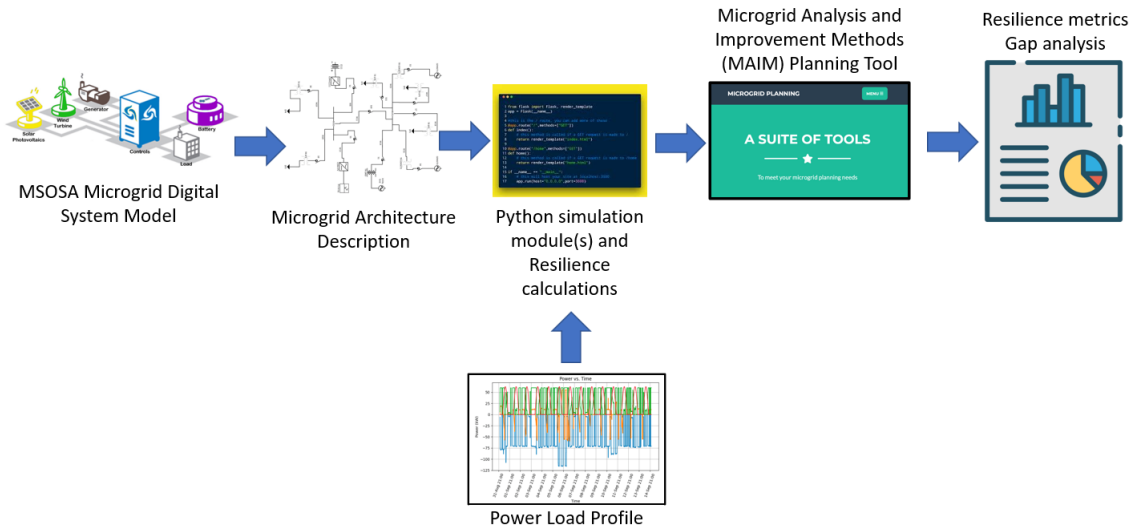


Figure 10. Engineer Operational Vignette. Adapted from TONEX (n.d.) and Peterson et al. (2021).

## F. USE CASES

Informed by the operational vignettes, the team continued to use Innoslate to develop a use case diagram and supporting action diagrams. This use of an MBSE tool provides all the same benefits as the traditional SE models demonstrate in functional analysis. Use cases were broken down into action diagrams, and action blocks in these diagrams were linked to requirements to ensure addressing of stakeholder needs.

Figure 11 is the high-level use case diagram for the Microgrid Resilience Analysis Software Development project MPT enhancements. Based on information gained from composing the operational vignettes and soliciting stakeholder feedback, nine primary use cases were generated, and the two operators added to show interaction points. As this use case diagram is a reflection of the project vision, not all use cases were targeted for completion for this project. For this capstone project, the use cases in blue circles (UC2, UC3, UC6, and UC7) were selected for development with concurrence from both stakeholders and advisors. UC8, in green, indicates functionality already existing in the MPT software suite that the team leveraged as its development foundation. Use cases in white (all others) were not selected for development.

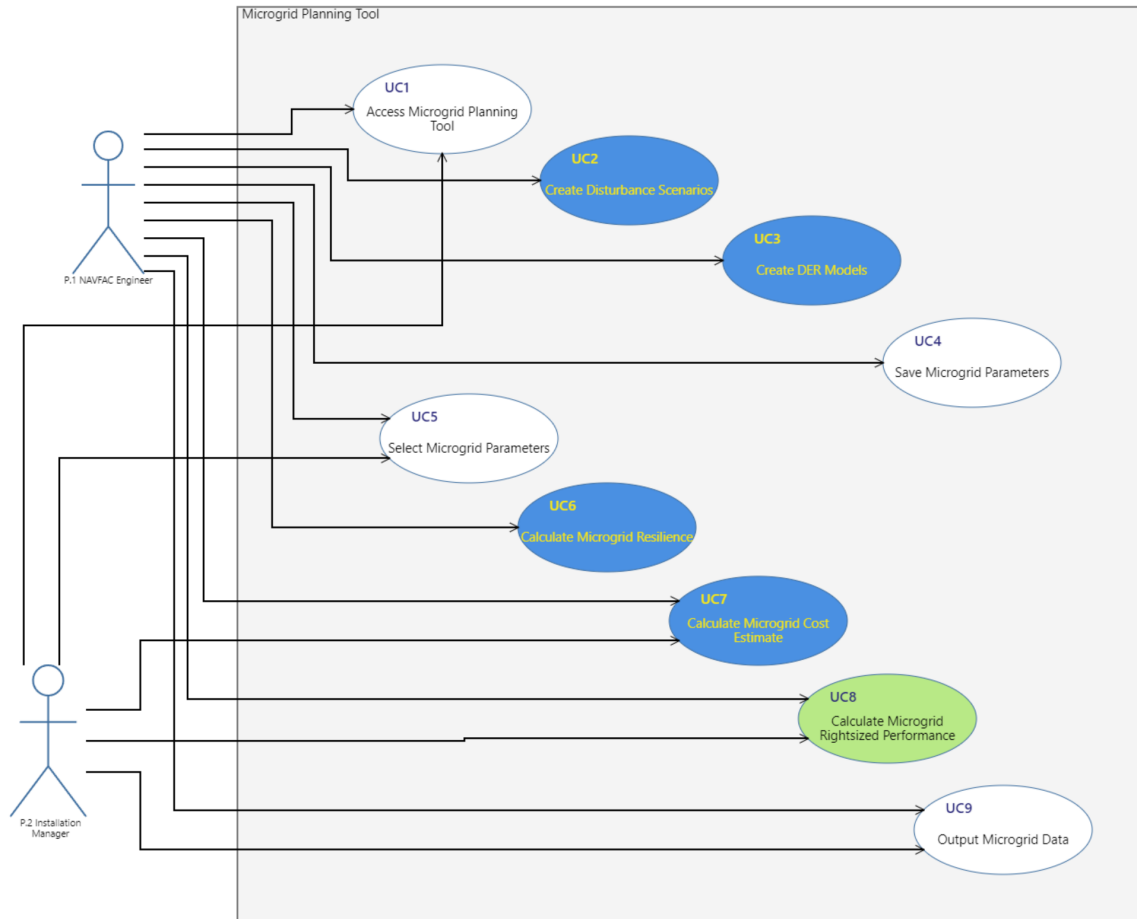


Figure 11. Microgrid Resilience Analysis Software Development Use Case Diagram

### 1. Calculate Microgrid Resilience (Use Case 6, MVP)

The “Calculate Microgrid Resilience” use case action diagram is represented by Figure 12 with inputs, outputs, and actions once again illustrated in similar fashion as in Figure 13 and Figure 14. This use case was selected to be the minimum viable product of this work, as it encapsulates the core thrust of the project’s technical goals. In this activity flow, the process begins with the user specifying the number of simulation runs and details of the microgrid simulation scenario that he/she wishes to execute. Each individual simulation run then determines particular DERs that are damaged and establishes recovery times for the damaged DERs based on associated probability distributions. Each run then generates data for power flow, load balancing, and battery charge levels across the

microgrid, considering the damaged and recovering DERs, over the duration of a defined time period. After the input number of simulation runs have been executed, the software calculates and returns resilience metrics for the entire series.

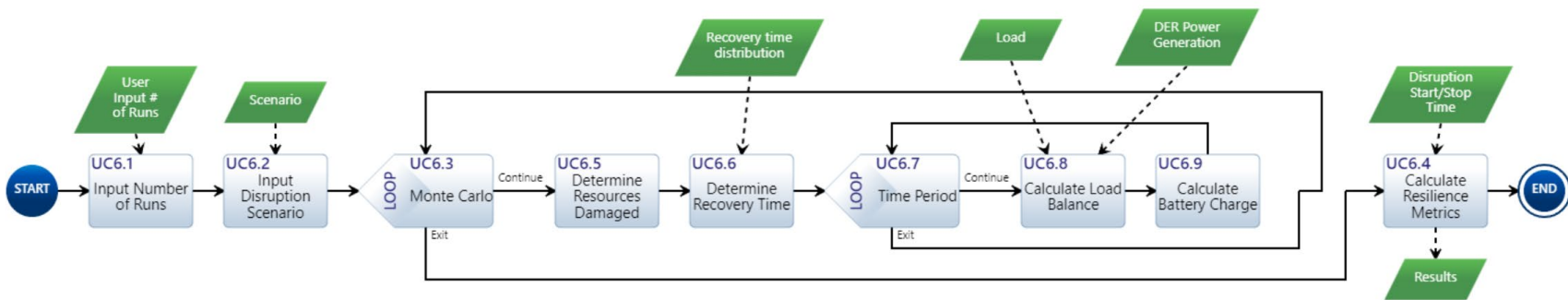


Figure 12. UC6 Calculate Microgrid Resilience Action Diagram

## 2. Create Disturbance Scenarios (Use Case 2)

Figure 13 displays the action diagram for the “Create Disturbance Scenarios” use case, which summarizes the process of identifying and selecting available microgrid disturbance scenarios, and then executing a microgrid simulation involving the selected disturbance. Inputs and outputs are represented by the green boxes, and discrete actions by the white boxes. First, files describing pertinent aspects of the simulatable disturbances (constructed in JavaScript Object Notation, or JSON, format), are loaded by the software API. Next, the user interacts with the API by issuing an HTTP GET request (via some interactive method, such as a web application or Jupyter notebook) to select the desired disturbance scenario, instructing the software to load the corresponding JSON file. Finally, the microgrid simulation is performed, including the selected disturbance type, and the resilience analysis results are output. This use case implicitly includes the ability for technical users to create, load, select, and execute their own disturbance JSON files.

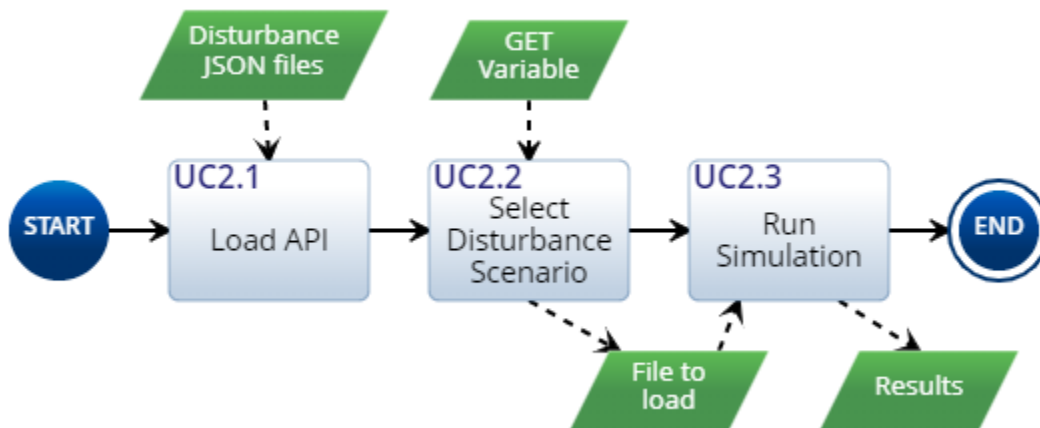


Figure 13. UC2 Create Disturbance Scenarios Action Diagram

## 3. Create DER Models (Use Case 3)

Figure 14 illustrates the “Create DER Models” use case action diagram, which follows the same input, output, and action portrayal conventions as Figure 13. This process begins with the software API again loading JSON files, though in this instance they define

and describe DERs that constitute pre-configured microgrid architecture models. The user then specifies whether they wish to use one of these existing options, or if they would rather create a new one, by providing either a true or false value for the “addnew” variable (this is representative of a more intuitive user interface element providing similar functionality). If a new addition is indicated as desired, the user then provides details of the microgrid DER architecture model to be created, reviews the composed JSON data detailing their specifications, then issues an HTTP GET request to the software API to select and implement the model that they just constructed. Alternatively, if the user instead decides to use one of the pre-existing models, they begin by obtaining and reviewing the JSON data for these architectures, then follow by issuing an API request pointing to their architecture selection. The activity flow ends with the user providing additional simulation parameters, executing the microgrid simulation (using either their created or selected DER composition model), and obtaining the output results.

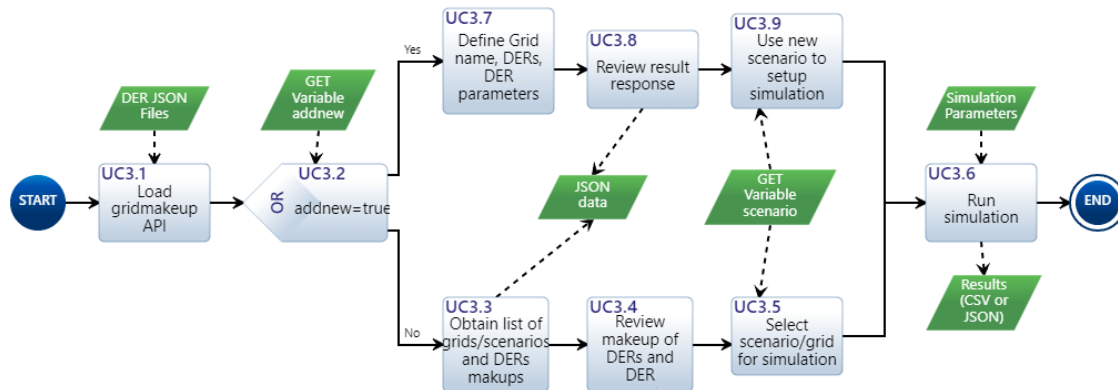


Figure 14. UC3 Create DER Models Action Diagram

#### 4. Calculate Microgrid Cost Estimate (Use Case 7)

Use case 7, “Calculate Microgrid Cost Estimate,” was eventually deferred due to project time constraints. The team realized early in the technical effort that this was a likely occurrence and communicated this to stakeholders. Given the choice of selecting which use case to drop, the decision was made to drop cost estimation as it had the least support existing in the MPT suite.

This occurrence clearly highlighted the advantages of the Agile SE process chosen by the team. When a delay in schedule occurred, the option of forgoing a product deliverable did not impact the functionality of the other use cases delivered.

## **G. SYSTEMS ENGINEERING MANAGEMENT PROCESSES**

To guide SE management for the project, a project management plan (PMP) was developed to define the initial SE approach, the stakeholder analysis, and problem statements. These have been covered in detail in previous sections. The SE PMP also covered the organizational structure of the team including roles and responsibilities. It defined how configuration management would be done, and identified what tools would be used to complete the work. Additionally, it discussed how the team would handle risk management and defined a schedule for the team as well.

### **1. Schedule**

Figure 15 shows the team schedule as a linear timeline. Sprints in green corresponded roughly to early team efforts including problem and need definition, stakeholder and requirements analysis, and generation of more traditional SE products. Beginning in the second quarter of the project, shown by the sprints in yellow, the team hit its stride with mapping sprints to use case deliverables. In each of sprints 4–9, a functional technical component of the resilience analysis software development was “delivered” to advisors and stakeholders, with demonstrations of functioning software at each sprint end. In the third quarter of the project, efforts turned toward processing simulation sensitivity analysis data, and delivering software artifacts and this report.



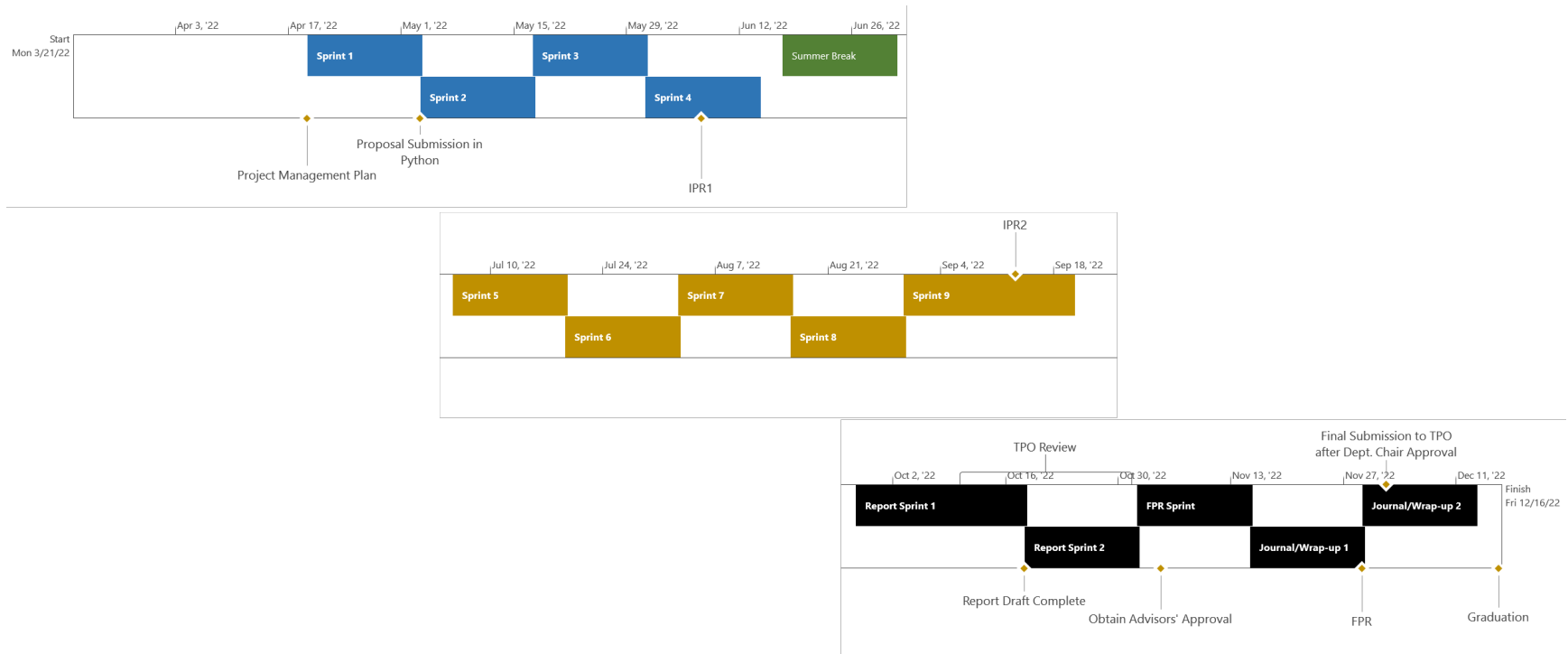


Figure 15. Project Schedule

## 2. Sprint Schedule Detail

Table 3 provides a listing of the 13 sprints that the team completed, including the sprint goals for each. Sprints were generally two weeks long, except for sprints at the end of quarters, which included additional work over the breaks. In general, a use case was sized to fit inside one sprint. This mostly held true, with the first selected use case (“Calculate Microgrid Resilience”) taking two sprints.

Table 3. Sprint Schedule List and Goals

<u>Sprint</u>	<u>Start Date</u>	<u>Goal</u>
1	4/19/22	Trace needs down to requirements and get stakeholder concurrence. Signed off PMP/CPF.
2	5/3/22	Write “Chapter 1” draft of capstone report and have advisors review and successfully execute Python microgrid planning tool analysis scripts.
3	5/17/22	Complete report chapter 2 “draft.” Map requirements to functional model. Complete IPR slides and rehearsal presentation.
4	5/31/22	Present IPR1. Create Sprint 5–10 tasks based on Project Vision and use case diagrams. Be prepared for Summer Quarter.
5	7/5/22	Calculate Microgrid Resilience. Create method that contains resilience math and matching test harness.
6	7/19/22	Return a resilience calculation based on a disturbance. Improve analysis narrative in report draft.
7	8/2/22	Implement Disturbance Scenario selection in web tool. Begin moving Resilience output to web tool.
8	8/16/22	API reports of Statistical resilience analysis. DER/grid selection via API. Work on Chapter 4 of Report
9	8/30/22	Build and present IPR 2. Fully integrate “AggregateMetrics” class with API. API cleanup. Ch 2/3 cleanup (Fall break: Finalize Python Notebook integration. Run Monte Carlo simulations.)
10	9/27/22	Finalize Merge software upstream with Dr. Reich. Complete Report Initial Draft to TPO
11	10/18/22	Incorporate TPO and advisor feedback. Final draft to SE Dept Chair
12	11/1/22	Create FPR slides and present FPR
13	11/15/22	Journal work and wrap up

### 3. Risks

The team used a standard risk matrix, considering the probability and impact of risks to the project, to generate risk ratings. All risks were considered medium at the beginning of the project, and with mitigations via control, were eventually reduced to low.

Risk for this project largely focused on inability to deliver a functioning product by the hard deadline imposed by NPS, mathematical errors in calculations, and availability of the project’s products. Due to the nature of the Agile SE processes, the risk of not completing the project was low. By utilizing use cases as units of work, we could ensure that each delivered use case was functional, tested, accurate, and stakeholder approved. While use cases could be missed due to deadlines, each use case did not depend on others to be fully functional. This is an ideal situation for this capstone effort. If this work is continued by other teams or researchers, there is no partially functional code delivered at the end of the project.

Risks were tracked in the project management plan; a graphical risk rating matrix is shown in Figure 16, and a summary of the risks is provided in Table 4.

<b>Impact:</b>		Minimal	Minor	Moderate	Significant	Severe
<b>Probability</b>	Very High					
	Likely					
	Possible				2 3	
	Unlikely				1 4	
	Very Unlikely					

Figure 16. Risk Rating Matrix

Table 4. Risk Summary

<u>ID</u>	<u>Title</u>	<u>Description</u>	<u>Type</u>	<u>Probability</u>	<u>Impact</u>	<u>Initial Rating</u>	<u>Mitigation Strategy</u>	<u>Revised Rating</u>
1	Project Delivery Timeline	If the microgrid resilience analysis software development timeline cannot be met, then the project's objectives will not be met.	Schedule	Unlikely	Significant	Medium	Control	Low
2	Microgrid Model Distributed Energy Resource (DER) Properties	If the DER attributes are not properly captured, then the resilience calculations will be inaccurate.	Performance	Possible	Significant	Medium	Control	Low
3	Resilience Calculation Validation	If the MPT resilience calculation additions are not validated, then their results will not provide value.	Performance	Possible	Significant	Medium	Control	Low
4	Tool Availability	If access to the MPT enhancements cannot be provided, then users will not be able to access them.	Performance	Unlikely	Significant	Medium	Control	Low

#### 4. Software Configuration Management

Software configuration management is a critical aspect of any software development project. Despite this project being relatively limited in scope and consisting of a small team, there were still many active developers working in a distributed environment. NPS offers access to a Git version control server, running GitLab to provide this capability. Along with necessary tools to merge and deconflict software, Git contains powerful attribution capabilities in the form of forking and commits.

The team chose to use Dr. Daniel Reich’s Microgrid Planning Tool Suite (Reich n.d.) as the foundation for implementing resilience calculations and metrics. Using Git, this was easy to do while maintaining attribution to the original project. The team forked the MPT software repository, as shown in Figure 17, to create a controlled downstream repository in which to develop the software further. At the end of development, it was then possible to merge the team’s changes back upstream to the original MPT suite project.

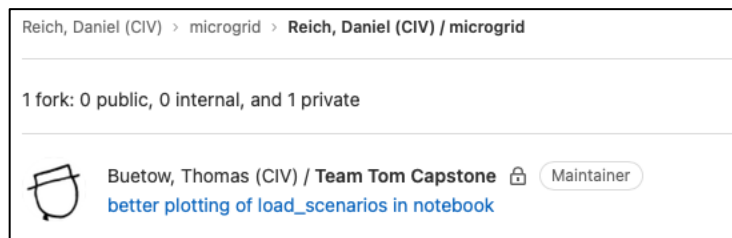


Figure 17. Fork of MPT Software Repository

The team followed a Git feature branch workflow for software development. This approach was selected due to its simplicity, ease of learning, and ability to keep the main branch functional at all times (Atlassian Bitbucket n.d.). In this workflow, developers on the team selected a feature to add to the codebase, made a branch, developed and tested the feature inside the branch, and then reviewed the changes with the team before merging the branch back into the main branch. This was repeated multiple times in a sprint until all features for the use case were added.

Git tracks individual developer contributions as commits. From a functional perspective, in the context of this project, commits are units of work related to meaningful changes to the branch. Commits can affect one or more lines in any number of files. The team reached a consensus on what constituted a “good” commit, and adhered to this convention during development. Figure 18 shows an excerpt of the GitLab UI tracking commits for this project.

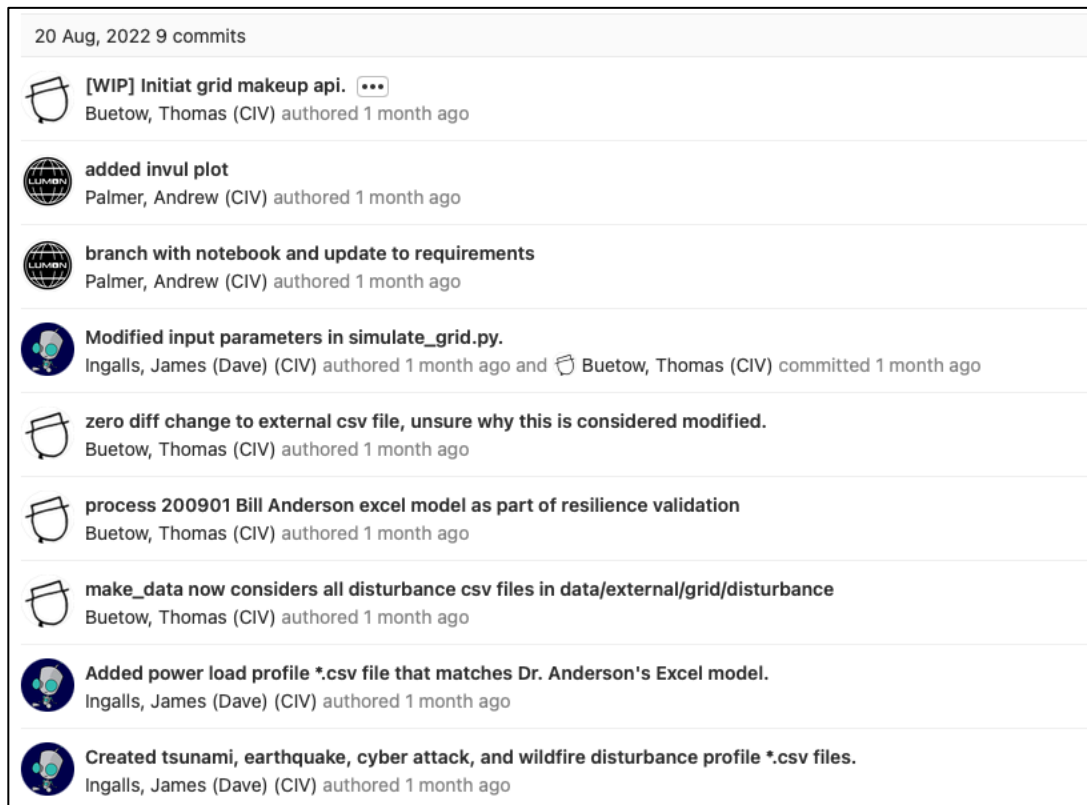


Figure 18. Software Commit Activity Excerpt

Once a feature, or set of features, was completed, the developer could initiate a merge request to move them into the master code base (in this instance referring to the primary branch of the team-specific repository). The team configuration manager, with team input, could review and integrate the software features into the master code base for other developers to use or incorporate into future additions. Each commit of software included traceability for each user, isolated to specific lines of code changed during the

commit process. As shown in Figure 19, GitLab enables side-by-side comparison of changes made during a commit prior to merging. This comparison allowed the team to review changes as needed, and identify specific changes made by each developer.

```
Changes 1
Showing 1 changed file with 12 additions and 3 deletions
Hide whitespace changes Inline Side-by-side
+12 -3 View file @1fc0ee2
src/models/simulate_grid.py
@@ -13,6 +13,7 @@ from src.grid import Grid
13 from src.data import defaults as data_defaults
14 from src.reports import Metrics, AggregateMetrics
15 from src.visualization import MetricPlots
16 + import re
17
18 class Simulation(object):
19
20 ... @@ -117,11 +118,19 @@ class Simulation(object):
117 i += 1
118 self._reset_original_loads()
119
120 - def scenario_to_csv(self):
121 -     csv = "time,load\n"
122
123     for s in self._scenarios:
124         delta = s.time_period().start() - self._start_datetime
125         csv += str(delta.total_seconds()/3600.0) + "," +
126         str(s.power_load()) + "\n"
127     return csv
128
129     def get_time_periods(self):
130
131 ... @@ -13,6 +13,7 @@ from src.grid import Grid
13 from src.data import defaults as data_defaults
14 from src.reports import Metrics, AggregateMetrics
15 from src.visualization import MetricPlots
16 + import re
17
18 class Simulation(object):
19
20 ... @@ -117,11 +118,19 @@ class Simulation(object):
118 i += 1
119 self._reset_original_loads()
120
121 + def scenario_to_csv(self, time_units="hours"):
122 +     if re.match('minute', time_units.lower()):
123 +         time_divisor = 60.0
124 +     elif re.match('second', time_units.lower()):
125 +         time_divisor = 1.0
126 +     else: #hours
127 +         time_units="hours"
128 +         time_divisor = 3600.0
129 +
130 +     csv = "time (" + time_units + "),load\n"
131     for s in self._scenarios:
132         delta = s.time_period().start() - self._start_datetime
133         csv += str(delta.total_seconds()/time_divisor) + "," +
134         str(s.power_load()) + "\n"
135     return csv
136
137     def get_time_periods(self):
138
```

Figure 19. Side-by-Side Comparison of a Commit in GitLab

Figure 20 illustrates how the branching process works. Time flows from bottom to top (older commits are at the bottom of the graph). The red line, furthest right, is the master (main) branch, and the teal and pink lines (second and third from right, respectively) are branches that the upstream project was working on simultaneously. This history exists due to the merging features of Git. The long running green line (fourth from right) is a testing branch from project team member Dave Ingalls. Beginning at the bottom of the graphic, at the red dot, team member Tom Buetow created two different branches from the same point in the code base, “resilience-to-web-MVP” (purple 5<sup>th</sup> from right) and “resilience-to-web” (yellow, furthest left). After the “resilience-to-web-MVP” feature was complete, it was merged into “Dave’s testing branch.” Later, “Dave’s testing branch” was merged back into

the master branch at the second red dot from the bottom. The purple “resilience-to-web” feature was then also merged into the master branch, but this copy of the master branch had itself diverged from the repository by a single commit (3<sup>rd</sup> red dot from bottom). An additional merge (top red dot) collapsed all branches back to master.

The configuration management capabilities of Git allow ease of tracking of where work was done, identification of who performed which work, and provides methods to revert work that was performed in error. As Git is a distributed system, all developers maintain full copies of the code base, ensuring that data loss is extremely unlikely.

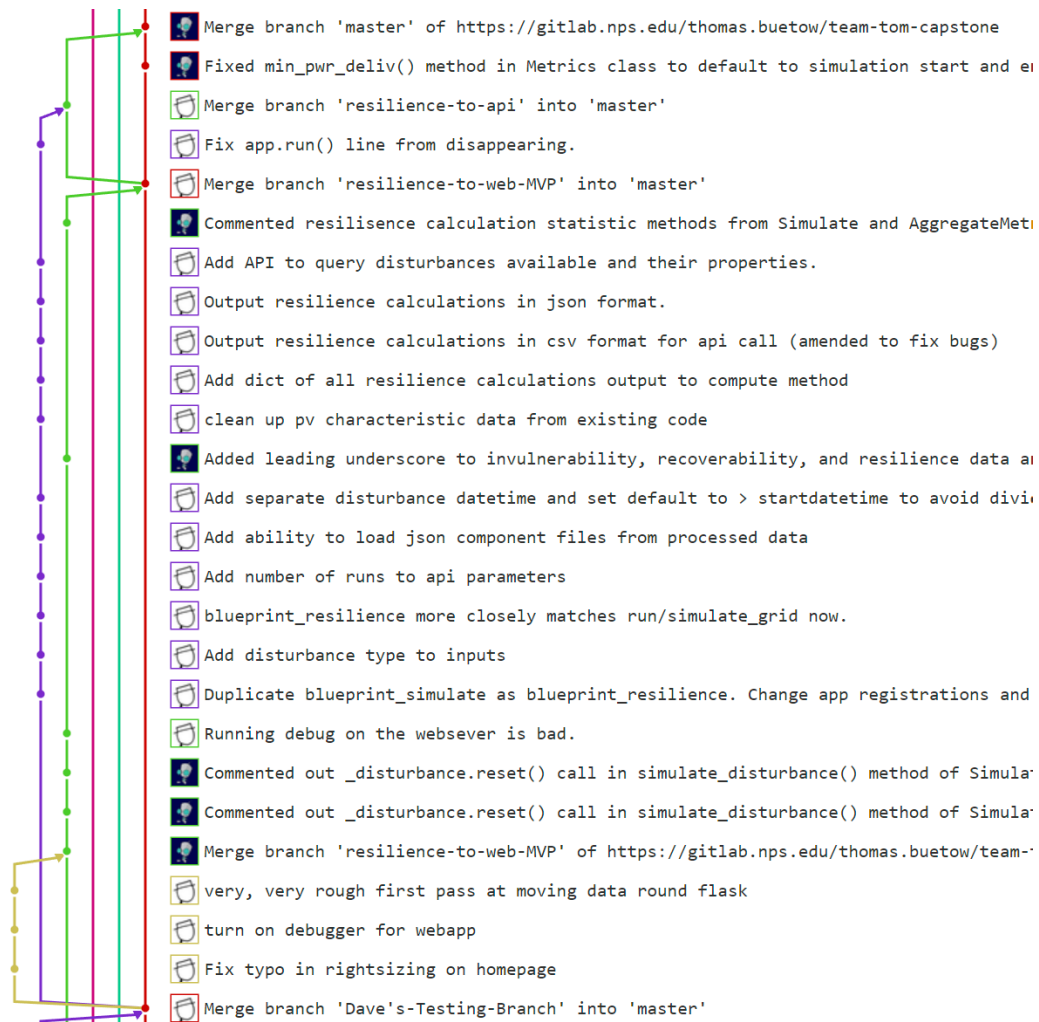


Figure 20. Microgrid Resilience Analysis Software Development Project Git Branches



THIS PAGE INTENTIONALLY LEFT BLANK

### III. SOFTWARE ARCHITECTURE AND DESIGN

#### A. FOUNDATIONAL SOFTWARE

The existing Microgrid Planning Tools (MPT) website (<https://microgrid.nsetti.nps.edu>) (Reich n.d.) provides a web-based interface to specify aspects of a simple microgrid architecture and scenario, including DER performance parameter values, pre-defined power load profiles, and selection of solar radiation profile based on geographic location (though, during the composition of this report, only one location was available). After configuring these input parameters, MPT will simulate the power flow of the microgrid over a duration dictated by the chosen power load profile, notably detailing battery charging and discharging activity, and periods where available microgrid power is insufficient to meet power demand. Figure 21 displays an example of configured DER parameters and the resultant output plots, illustrating details of the power flow simulation. MPT also offers a “rightsizing” algorithm and graphical output, guiding the selection of DER quantities for a microgrid. This rightsizing determination is based on the ability of DERs conforming to the user-specified performance criteria to meet the chosen power load, while simultaneously minimizing excess power, however this feature is outside the scope of this capstone effort.



Figure 21. Microgrid Planning Tools Power Flow Simulation Web Interface. Source: Reich (n.d.)

The code base for MPT is primarily written in the programming language Python, and is maintained in an NPS-administered GitLab repository (at the time of this writing, chiefly managed by Dr. Daniel Reich). A fork (distinct copy) of this repository was created, and utilized as the foundational software architecture from which to build upon and implement the selected resilience calculations, in order to meet the corresponding resilience analysis software system requirements. In collaboration with Dr. Reich, this fork (imaginatively and affectionately titled the “Team Tom Capstone Fork”) was validated and merged with the primary MPT repository, integrating the resilience analysis code augmentations into the main code base.

A comprehensive description of the MPT code architecture and functionality would far exceed the scope of this report. For further information, the reader is encouraged to reference the MPT application programming interface (API) documentation (<https://microgrid.nsetti.nps.edu/api>), and to contact the current MPT site administrator(s). Access to the code base may also be requested from the site administrator(s). For the

purposes of edifying code contributions made by this capstone team (detailed below), a brief explanation of pertinent aspects of the power flow simulation code follows.

The resilience calculation software development additions focused on three existing custom Python classes in the MPT code base. A custom “Simulation” class defines top-level data elements and methods (which, in turn, reference additional custom classes, for example classes defining the attributes and behaviors of power generator DERs) for performing microgrid power flow simulations. The time-series output data of any one simulation is stored in a custom “Metrics” class, organized into Python data dictionaries, providing a [key]:[value] data structure. Many of the keys and values of these “Metrics” class data dictionaries contain multiple elements and/or are hierarchical, composed of additional custom classes, and even further data dictionaries; though complex, this offers multiple avenues for key indexing and searching, assuming one knows what is being sought after. Lastly, a custom “AggregateMetrics” class stores the per-time-unit average power flow data of multiple simulations executed in series, and is otherwise structured effectively identically to the “Metrics” class.

## **B. RESILIENCE METRICS CALCULATION ADDITIONS**

Bekera and Francis (2014) posit system resilience as subject to three fundamental properties, similar to the ESF: absorptive capacity, adaptive capacity, and recovery capacity. This supposition broadly structured the team’s design approach. Specialized microgrid calculations were informed by prior analysis efforts which predicate the Python code. The following descriptions identify the equations of interest, provide their origin and reasoning, and discuss their integration into the MPT code base.

### **1. Invulnerability**

In order to specify microgrid absorptive and recovery capacity, the resilience model developed by Giachetti et al. (2022) was utilized. Absorptive capacity is the quality of withstanding perturbations with minor consequences. As shown in (1), the ratio of delivered to demanded power informs invulnerability.

*I: Invulnerability*

$$I = \frac{P_t}{D_t}$$

$P_t$  : power delivered at time  $t$  (1)

$D_t$  : power demanded at time  $t$

$$P_t \leq D_t$$

$$I \in [0,1]$$

Of note, it is important to differentiate power generation capacity and power delivered to the load. Microgrids are constructed to operate in islanding scenarios, consequently generation capacity ought to surpass demand (Giachetti et al. 2022). Since delivered power cannot exceed demand, the maximum invulnerability value achievable is unity.

When implementing the invulnerability calculation into the MPT code, a challenge arose in identifying singular values for “power delivered” and “power demanded” ( $P_t$  and  $D_t$  respectively, in (1)). This is illustrated with help from Figure 22.

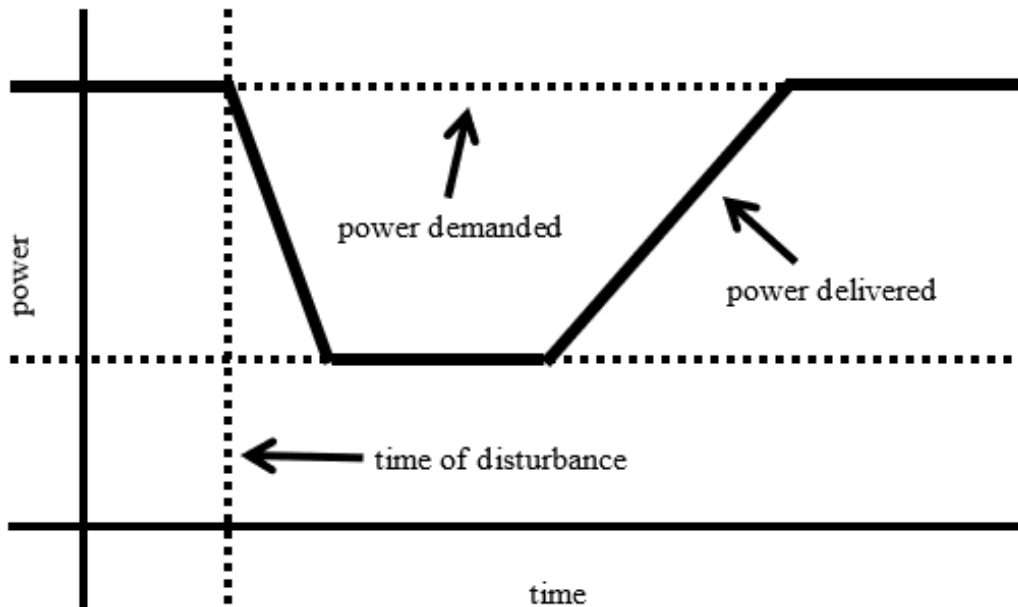


Figure 22. Ideal Power Flow Curve with Disturbance

In the ideal power flow curve, the values for  $P_t$  and  $D_t$  are trivial to identify. The power demanded ( $D_t$ ) is a static, constant value at the top of the curve, and the power delivered ( $P_t$ ) is the flat bottom of the curve after the disturbance has occurred. Unfortunately, more realistic power flow data resembles that shown in Figure 23.

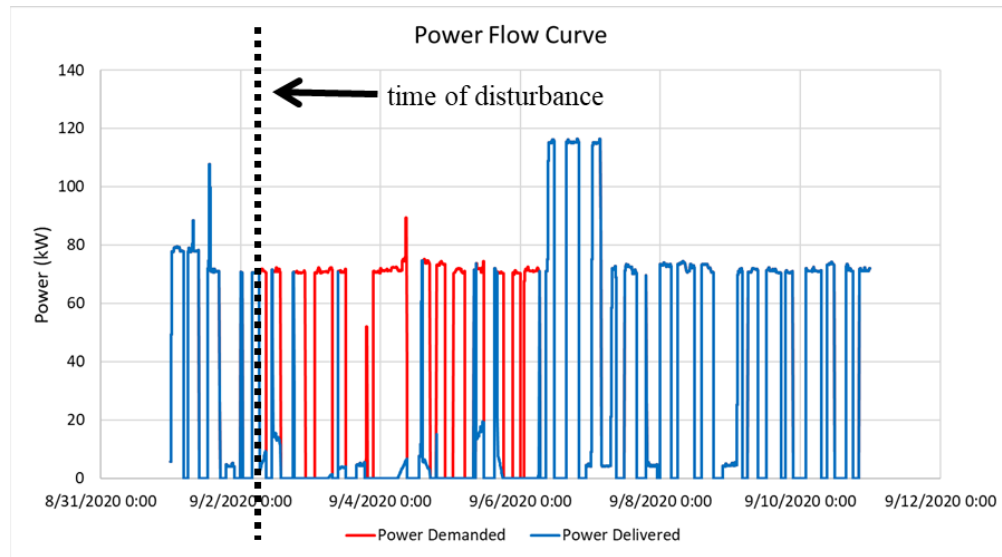


Figure 23. Simulated Power Flow Curve with Disturbance

In this simulated power flow data, the power demanded fluctuates significantly over time, dropping to near-zero during most night-time periods, then quickly rising back up to a non-constant elevated value through most day-time periods. Because the intent of the invulnerability calculation is to quantify a microgrid’s absorptive capacity against a single, consistent “power demanded” value, a decision was presented in how to reduce the multiple time-period-specific pre-disturbance power demanded values down to a single one, while still keeping this value reflective of the actual power demanded by the simulated microgrid. The enacted solution was to construct a framework for implementing and selecting from multiple calculation methods for determining this simplified power demanded value, and within which a couple of already-developed methods would be provided as standard. These default methods consist of an averaging calculation, which calculates and utilizes the average power demanded prior to the disturbance; and a

maximum calculation, which determines and returns the maximum single power demanded value prior to the disturbance. Though a similar multiple-to-single-value issue is encountered for the power delivered value, as invulnerability is a measure of the microgrid's worst-case performance, the solution was merely to determine the minimum single power delivered value over the course of the disturbance's impact to the microgrid. Python code which implements this logic and the resultant invulnerability calculation is shown in Figure 24, with definitions provided for the input variables in Figure 25.

```

# initialize power demanded list
p_demand = []
# initialize disturbance record count
pre_dist_rec_cnt = 0

# begin iterating through all simulation records, earliest to latest
for i in range(len(sim_keys)):

    # check if current record is before disturbance start time
    if (sim_keys[i].time_period().start() < t_start):

        # determine power demanded for current record
        cur_p_demand = abs(self.power[sim_keys[i]]["Load"])
        # append power demanded to list
        p_demand.append(cur_p_demand)
        # increment pre-disturbance record count
        pre_dist_rec_cnt = pre_dist_rec_cnt + 1

    # check if current record is beyond disturbance start time
    if (sim_keys[i].time_period().start() >= t_start):
        # exit for loop
        break

# check selected pre-disturbance power demanded calculation
# methodology selection
if (pre_dist_calc_key == "max"):

    # calculate maximum power demanded
    p_demand_val = max(p_demand)

else:

    # calculate average power demanded
    p_demand_val = sum(p_demand) / pre_dist_rec_cnt

# calculate and return invulnerability
invul = p_deliv_min / p_demand_val

```

Figure 24. Invulnerability Calculation Python Code

```

# sim_keys: a list of keys for the data dictionary that stores the
#   simulated power values
# t_start: disturbance start time
# self: the method-containing class, which also contains the data
#   dictionary that stores the simulated power values
# pre_dist_calc_key: a string indicating which pre-disturbance power
#   demanded calculation method should be utilized
# p_deliv_min: the minimum power delivered value during the course
#   of the disturbance's impact to the microgrid

```

Figure 25. Input Variable Definitions for Invulnerability Calculation Python Code

Figure 26 and Figure 27 detail Python code that determines the “p\_deliv\_min” value utilized in the invulnerability calculation code shown in Figure 24. Input variables to this code are defined in Figure 28.

```

# search through all simulation records, from earliest to latest
for i in range(len(sim_keys)):
    # check if current record datetime is equal to search start
    # datetime
    if (sim_keys[i].time_period().start() == t_start):
        # exit for loop
        break

# set power delivered and power demanded values for current record
cur_p_record = self.power[sim_keys[i]]
cur_p_deliv = 0

for gen_key in cur_p_record.keys():
    if (gen_key != "Load"):
        cur_p_deliv += cur_p_record[gen_key]

cur_p_demand = abs(cur_p_record["Load"])

# check that power delivered does not exceed power demanded
if (cur_p_deliv > cur_p_demand):
    # set power delivered to power demanded
    cur_p_deliv = cur_p_demand

# initialize minimum power delivered value
p_deliv_min = cur_p_deliv

```

Figure 26. Minimum Power Delivered During Disturbance Impact Determination Python Code (1/2)



```

# search through all simulation records, from beginning of
# disturbance onward
for j in range(i, len(sim_keys)):

    # check if current record datetime is within search start and end
    # datetimes
    if ((sim_keys[j].time_period().start() >= t_start) and
        (sim_keys[j].time_period().start() <= t_end)):

        # set power delivered and power demanded values for current
        # record
        cur_p_record = self.power[sim_keys[j]]
        cur_p_deliv = 0

        for gen_key in cur_p_record.keys():
            if (gen_key != "Load"):
                cur_p_deliv += cur_p_record[gen_key]

        cur_p_demand = abs(cur_p_record["Load"])

        # check that power delivered does not exceed power demanded
        if (cur_p_deliv > cur_p_demand):
            # set power delivered to power demanded
            cur_p_deliv = cur_p_demand

        # check if power delivered value for current record is smaller
        # than previously identified minimum power delivered
        if (cur_p_deliv < p_deliv_min):
            # set new minimum power delivered value
            p_deliv_min = cur_p_deliv

    # check if current record is beyond disturbance end time
    if (sim_keys[j].time_period().start() > t_end):
        # exit for loop
        break

```

Figure 27. Minimum Power Delivered During Disturbance Impact Determination Python Code (2/2)

```

# sim_keys: a list of keys for the data dictionary that stores the
# simulated power values
# t_start: disturbance start time
# self: the method-containing class, which also contains the data
# dictionary that stores the simulated power values
# t_end: disturbance end time

```

Figure 28. Input Variable Definitions for Minimum Power Delivered During Disturbance Impact Determination Python Code

## 2. Recoverability

Recoverability is the characterization of a microgrid's restoration to nominal operating conditions after a disruption. The microgrid is said to be in a recovered state when power delivered returns to a sufficient level to meet power demanded. The recoverability expression, (2), relates the integration curves of the power differential and the power demand (Giachetti et al. 2022). The recoverability calculation is only meaningful when  $P_t$  is less than  $D_t$ .

$$\begin{aligned} R: & \text{Recoverability} \\ R = & 1 - \frac{\sum_{t=t_d}^{t_r} D_t - P_t}{\sum_{t=t_d}^{t_r} D_t} \\ t_d: & \text{time of disturbance} \\ t_r: & \text{time of full recovery} \\ P_t: & \text{power delivered at time } t \\ D_t: & \text{power demanded at time } t \\ R \in & [0,1] \end{aligned} \tag{2}$$

Since recoverability is essentially a ratio of integration calculations, it is straightforward to implement in Python code, depicted in Figure 29. The input variables utilized in this code are defined in Figure 30.

```

# initialize power demanded delivered difference and power demanded
# lists
p_demand_deliv_diff = []
p_demand = []

# begin iterating through all simulation records, earliest to latest
for i in range(len(sim_keys)):

    # check if current record is within disturbance start and end
    # times
    if ((sim_keys[i].time_period().start() >= t_start) and
        (sim_keys[i].time_period().start() <= t_end)):

        # determine power demanded and power delivered for current
        # record
        cur_p_record = self.power[sim_keys[i]]
        cur_p_deliv = 0

        for gen_key in cur_p_record.keys():
            if (gen_key != "Load"):
                cur_p_deliv += cur_p_record[gen_key]

        cur_p_demand = abs(cur_p_record["Load"])

        # check that power delivered does not exceed power demanded
        if (cur_p_deliv > cur_p_demand):
            # set power delivered to power demanded
            cur_p_deliv = cur_p_demand

        # append power demanded delivered difference and power delivered
        # for current record to appropriate lists
        p_demand_deliv_diff.append(cur_p_demand - cur_p_deliv)
        p_demand.append(cur_p_demand)

    # check if current record is beyond disturbance end time
    if (sim_keys[i].time_period().start() > t_end):
        # exit for loop
        break

# calculate and return recoverability
recov = 1 - (sum(p_demand_deliv_diff) / sum(p_demand))

```

Figure 29. Recoverability Calculation Python Code

```

# sim_keys: a list of keys for the data dictionary that stores the
#   simulated power values
# t_start: disturbance start time
# t_end: disturbance end time
# self: the method-containing class, which also contains the data
#   dictionary that stores the simulated power values
# p_deliv_min: the minimum power delivered value during the course
#   of the disturbance's impact to the microgrid

```

Figure 30. Input Variable Definitions for Recoverability Calculation Python Code

### 3. Resilience

Resilience is a composite metric, combining both invulnerability and recoverability, as shown in (3) (Giachetti et al. 2022). The weighting factor,  $\omega$ , reflects the investigation goals of the decision maker. This method allows isolation of invulnerability or recoverability when inspecting at the boundary conditions of  $\omega$ . Additionally, importance can be distributed as required.

$$\begin{aligned}
 &\xi: \text{Resilience} \\
 &\xi = \omega I + (1 - \omega)R \\
 &\omega: \text{weighting factor} \\
 &\omega \in [0,1] \\
 &\xi \in [0,1]
 \end{aligned} \tag{3}$$

Having previously determined values for invulnerability and recoverability using the code detailed in Figure 24 and Figure 29, respectively, resilience is easily computed using the Python code in Figure 31, with input variables defined in Figure 32.

```

# calculate and return resilience
resil = (omega * invul) + ((1 - omega) * recov)

```

Figure 31. Resilience Calculation Python Code

```
# omega: weighting factor
# invul: invulnerability
# recov: recoverability
```

Figure 32. Input Variable Definitions for Resilience Calculation  
Python Code

#### 4. Supporting Functions

As the disturbance start and stop times are input parameters for the invulnerability and recoverability calculations, the output values of these functions, and ultimately the calculated resilience value, are dependent upon these times. In instances where the time when a disturbance is initiated is unknown, it may be estimated by observing when the power delivered drops significantly below the power demanded. Similarly, the time when a disturbance ends may be assessed as the time when power delivered returns to match the power demanded. However, for recoverability in particular, defining the disturbance start and end times in such a fashion returns a worst-case calculated recoverability value (and hence, an associated worst-case resilience value), since it will result in the numerator of the second term having no summands with a non-zero difference between power demanded and power delivered, a minimum-value denominator, and an overall maximum subtrahend from the ideal recoverability value of 1. If the disturbance start and end times are, instead, set to times “outside” of the duration of adverse impact to microgrid power delivered, there will be an increase in zero-value numerator summands, as well as an increase in the denominator value, improving the calculated recoverability and resiliency values.

To allow for either circumstance, the respective methods containing the invulnerability, recoverability, and resilience code, shown in Figure 24, Figure 29, and Figure 31, allow either for specific defined disturbance start and end times to be passed in, or else will call a method which determines these times based on the methodology described in the previous paragraph. Figure 33 shows the conditional structure included at the beginning of each of the invulnerability, recoverability, and resilience methods, which checks if these times have been provided, and otherwise invokes the disturbance-time-finding methods. The input variables for this code are defined in Figure 34.

```

# check if start time of disturbance provided
if (t_start == None):
    # determine datetime for simulation record at start of disturbance
    t_start = self.dist_start_time(p_dist_thresh)

# check if end time of disturbance provided
if (t_end == None):
    # determine datetime for simulation record at end of disturbance
    t_end = self.dist_end_time(p_dist_thresh)

```

Figure 33. Disturbance Start and End Time Determination Conditional Structure Python Code

```

# t_start: disturbance start time (if initially equal to "None", no
#   defined start time has been provided
# t_end: disturbance end time (if initially equal to "None", no
#   defined end time has been provided
# self: the method-containing class, which also contains the data
#   dictionary that stores the simulated power values
# p_dist_thresh: the minimum power deficit (power delivered
#   subtracted from power demanded) that must be present for the
#   simulated microgrid to be considered impacted by a disturbance
# dist_start_time(): a method which returns the first time index of
#   a microgrid power flow simulation where the power deficit is
#   greater than or equal to the "p_dist_thresh" value passed in
# dist_end_time(): a method which returns the last time index of a
#   microgrid power flow simulation where the power deficit is
#   greater than or equal to the "p_dist_thresh" value passed in

```

Figure 34. Input Variable Definitions for Disturbance Start and End Time Determination Conditional Structure Python Code

Figure 35 displays the pertinent code for the “dist\_start\_time()” method, with input variables defined in Figure 36. This method iterates forward in simulated time through the time indices, searching for the first instance where the difference between the demanded and delivered power values (the power deficit) exceeds the provided threshold value. The “dist\_stop\_time()” method is nearly identical, however it iterates backwards in simulated time, thereby identifying the last instance where the power deficit exceeds the provided threshold value (the “range()” function is provided with alternate arguments to achieve the reverse-search).

```

# initialize start time value
t_start = None

# search through all simulation records, from earliest to latest
for i in range(len(sim_keys)):

    # check if power deficit for current record exceeds threshold
    if (self.excess_deficit[sim_keys[i]] < thresh):

        # set new start time value
        t_start = sim_keys[i].time_period().start()
        # exit for loop
        break

# return start time of disturbance
return(t_start)

```

Figure 35. Disturbance Start Time Search Python Code

```

# sim_keys: a list of keys for the data dictionary that stores the
# simulated power values
# self: the method-containing class, which also contains the data
# dictionary that stores the simulated power values
# thresh: the minimum power deficit (power delivered subtracted from
# power demanded) that must be present for the simulated microgrid
# to be considered impacted by a disturbance

```

Figure 36. Input Variable Definitions for Disturbance Start Time Search Python Code

In addition to providing resilience metric calculation capability for individual power flow simulation runs, the utility of offering basic statistical analyses of these metrics across multiple simulations in a series was also recognized. Because unique power generator DER impact determination, repair/restoration times, and solar irradiance are randomized for each single simulation, successive simulations with otherwise identical defining input and control parameters will likely return differing resilience metric values. Thus, code was added to store the calculated invulnerability, recoverability, and resilience values from each individual simulation run of a series to a Python list, then to calculate the mean and standard deviation of each of these lists after execution of the last simulation of the series. These statistical values are then available for further investigatory efforts, for example to inform a sensitivity analysis between differing microgrid architectures.

Figure 37 displays the lines of Python code executed after each individual simulation run, appending single-simulation-run resilience metric values to respective appropriate Python lists, and Figure 38 displays the code which calculates the statistical means and standard deviations of the values stored in these lists. Input variables for these code excerpts are defined in Figure 39. Note the conditional check for an existing value in the “dist\_t\_start” variable, determining whether a definitive disturbance start time value is passed into the resilience metric calculation methods, or if a search for a disturbance start time is conducted, as described in the paragraph prior to Figure 35.

```
# determine individual simulation run data values and append to
# appropriate lists
if (dist_t_start):

    self.invol_list.append(metrics[0].calc_invol(t_start =
        dist_t_start))
    self.recov_list.append(metrics[0].calc_recov(t_start =
        dist_t_start))
    self.resil_list.append(metrics[0].calc_resil(t_start =
        dist_t_start))

else:

    self.invol_list.append(metrics[0].calc_invol())
    self.recov_list.append(metrics[0].calc_recov())
    self.resil_list.append(metrics[0].calc_resil())
```

Figure 37. Single Simulation Run Resilience Metrics Value List Storage Python Code

```
# calculate mean and standard deviation for invulnerability,
# recoverability, and resilience
self.invol_mean = sum(self.invol_list) / len(self.invol_list)
self.invol_std_dev = (sum([(x - self.invol_mean) ** 2)
    for x in self.invol_list]) / len(self.invol_list) ** 0.5
self.recov_mean = sum(self.recov_list) / len(self.recov_list)
self.recov_std_dev = (sum([(x - self.recov_mean) ** 2)
    for x in self.recov_list]) / len(self.recov_list) ** 0.5
self.resil_mean = sum(self.resil_list) / len(self.resil_list)
self.resil_std_dev = (sum([(x - self.resil_mean) ** 2)
    for x in self.resil_list]) / len(self.resil_list) ** 0.5
```

Figure 38. Resilience Metrics Values Statistical Calculations Python Code



```

# dist_t_start: disturbance start time
# self: the method-containing class, which also contains the lists
#   that store individual-simulation-run resilience metrics values,
#   and the class variable which contains the data dictionary that
#   stores the simulated power values
# invul_list: the list storing individual-simulation-run
#   invulnerability values
# recov_list: the list storing individual-simulation-run
#   recoverability values
# resil_list: the list storing individual-simulation-run resilience
#   values
# metrics: the class variable which contains the data dictionary
#   that stores the simulated power values
# calc_invul(): the invulnerability-calculating method
# calc_recov(): the recoverability-calculating method
# calc_resil(): the resilience-calculating method

```

Figure 39. Input Variable Definitions for Single Simulation Run Resilience Metrics Value List Storage and Resilience Metrics Values Statistical Calculations Python Code

### C. APPLICATION PROGRAMMING INTERFACE AUGMENTATIONS

Ideally, access to all of the highlighted code additions would be provided through the existing MPT web-based graphical user interface. However, this interface is constructed mainly in HTML and JavaScript code, familiarization with which would have been unfeasible in addition to already-committed efforts, given the time and resource constraints of the capstone team. In lieu of this, augmentations were effected to the MPT API, which offers access to MPT functions and output data via parameters passed by appending appropriate text strings to the MPT uniform resource locator (URL) web address. This affords follow-up programming activities conducted by future developers a pre-constructed means for accessing this data, and eventually incorporating it into a more user-friendly front end interface.

The programming contributions enabling these API additions are expansive and somewhat cryptic, comprised of over 400 lines of Python code largely referencing other elements of the MPT code base. For brevity, full reproductions and traceability explanations of these code elements are omitted from this report, however readers are encouraged to review the newly-created “api\_helpers.py,” “blueprint\_disturbance.py,” “blueprint\_gridmakeup.py,” and “blueprint\_resilience.py” files, and the augmented

“app.py,” “blueprint\_load.py,” “params.py,” and “openapi.yaml” files, in the “notebooks” and “webapps/api” directories of the Team Tom Capstone MPT code additions for complete details. Additionally, dynamic Jupyter Notebooks providing interactive example code blocks have been composed, and are available in the “notebooks” directory of the MPT code additions (these files have an extension of “ipynb”). Concise descriptions and examples of major enhancements to the base MPT API are provided below.

Of primary importance, a custom “MPT\_API\_QUERY” Python class was created which contains several data elements and methods enabling Python-based (as opposed to URL-based) API queries of microgrid simulation input parameter selections and details, and initiations of microgrid power flow simulation series. This provides additional flexibility to users and future developers in how they choose to implement and access the MPT API. In the API utilization examples that follow, both a URL-based and Python-based query/initiation of the demonstrated operation will be provided (the latter implementing methods included in the “MPT\_API\_QUERY” class). API responses are provided in JavaScript Object Notation, or JSON, format. Note that, for these examples, the API is being executed on a local machine instance (hence the “localhost” portion of the web address included in the URL), and portions of the returned JSON data are collapsed for succinctness (as indicated by the “jumps” in line numbers shown to the left, and the left/right arrow symbols shown to the right, of certain lines of the response data).

To inform users of the available MPT simulation configuration options, the API can be requested to return a list of pre-defined microgrid DER architectures, available power load profiles, and the types and characteristics of available microgrid disturbance events, as shown in Figure 40, Figure 41, and Figure 42. Figure 42 also includes a partial expansion of the compositional and performance details included with the returned microgrid architectures. Further information for querying and specifying parameters of microgrid power flow simulation runs via the API are provided directly on the API documentation website (for URL-based access; <https://microgrid.nsetti.nps.edu/api>), as well as within the previously-referenced Jupyter Notebooks included with the Team Tom Capstone MPT code additions (for Python-based access). A copy of the API documentation is provided in Appendix D.

Though comparatively small in magnitude of effort to construct and incorporate, it is worth mentioning that four additional microgrid disturbance event types (cyber attack, earthquake, tsunami, and wildfire) were added to the two originally included with MPT (hurricane and municipal outage). The probabilities of these disturbances to impact specific DERs were defined based on research by Anderson (2020). Additional, and significantly more involved, new inclusions into the API, include the ability to define specific quantities of power generator DERs and their performance parameter values (as opposed to selecting from a list of pre-configured microgrid architectures), the ability to cache the results of a simulation series (such that they are immediately retrieved if a series of identical input parameter specifications is requested to be executed), and an automatic power load profile extending algorithm, such that power load data of a limited duration can be appropriately expanded to (repeated across) any desired simulation timeframe. Further, the web-based API documentation has been comprehensively updated to detail all facets of the new inclusions.

URL:

http://localhost:5001/api/gridmakeup/json/

Python:

```
MPT_API_QUERY().api_endpoint("/api/gridmakeup/json")
```

Response:

```
4   {
5   "data": {
6     "air_terminal_1": [
7       {
8         "id": "m1",
9         "type": "MunicipalPower",
10        "attributes": {}
13      },
14      {
15        "id": "pv1",
16        "type": "SolarPhotovoltaicPanel",
17        "attributes": {
18          "power_rating": 250.0,
19          "economic_lifespan": 1.0,
20          "investment_cost": 0.0,
21          "annual_om_cost": 0.0,
22          "capacity": {},
36          "sine_width": {}
50        }
51      },
52      {
53        "id": "dg1",
54        "type": "DieselGenerator",
55        "attributes": {}
66      },
67      {
68        "id": "b1",
69        "type": "Battery",
70        "attributes": {}
83      }
84    ],
85    "air_terminal_2": [],
164   "air_terminal_2b": [],
228   "air_terminal_2c": [],
269   "air_terminal_3": [],
341   "air_terminal_3a": [],
413   "air_terminal_3_dup": [],
555   "air_terminal_3_quadruple": [],
627   "resilience_cost_201016": [],
840   "resilience_cost_201108": []
1089  }
1090 }
```

Figure 40. API Example of Querying Pre-defined Microgrid Composition

```
URL:
http://localhost:5001/api/load/json/

Python:
MPT_API_QUERRY().api_endpoint("/api/load/json")

Response:
4  ▾  {
5  ▾    "data": [
6      "air_terminal",
7      "air_terminal_2",
8      "air_terminal_2_anonymized",
9      "air_terminal_3_anonymized",
10     "air_terminal_4_anonymized",
11     "resilience_cost_201016",
12     "resilience_cost_201108"
13   ]
14 }
```

Figure 41. API Example of Querying Available Power Load Profiles

```
URL:
http://localhost:5001/api/disturbances/json/

Python:
MPT_API_QUERRY().api_endpoint("/api/disturbances/json")

Response:
4  ▾  {
5  ▾    "data": {
6      "cyber_attack": [↔],
78   "earthquake": [↔],
150  "hurricane": [↔],
222  "municipal_outage": [↔],
294  "tsunami": [↔],
366  "wildfire": [↔]
438  }
439 }
```

Figure 42. API Example of Querying Available Microgrid Disturbances

Armed with knowledge of the parameters that may be used to specify aspects thereof, utilizing the augmented MPT API to execute a microgrid power flow simulation series, and generating the resultant resilience metrics and statistical data, becomes merely an exercise in either constructing the appropriate URL string, or composing and passing the correctly assembled parameter-defining Python data dictionary. Again, details are included in both the web-based API documentation, and the Python-based Jupyter Notebooks. Figure 43 provides a simple example of initiating a series of 10 simulation runs using the “air\_terminal\_2” microgrid architecture, the “air\_terminal\_3\_anonymized” power load profile, and the “hurricane” disturbance. All other undefined parameters are left at their default values. Note the “sim\_data\_output” parameter, set to a value of “false,” prevents the return of the entire set of simulated discrete time index power flow data, vastly reducing the amount of information transmitted.

URL:

```
http://localhost:5001/api/resilience/json/  
?sim_data_output=false&num_runs=10  
&scenario_components=air_terminal_2  
&scenario_load=air_terminal_3_anonymized  
&disturbance=hurricane
```

Python:

```
query.params_as_dict({  
    "sim_data_output":False,  
    "num_runs":10,  
    "scenario_components":"air_terminal_2",  
    "scenario_load":"air_terminal_3_anonymized",  
    "disturbance":"hurricane",  
})
```

Response:

```
4  ▾ {  
5  ▾  "data": {  
6      "simruntime": "2022/10/02 16:44:57",  
7      "parameters": {  
40 ▾  "aggregate_resilience_metrics": {  
41      "minimum_power_load": -0.019999999999999997,  
42      "disturbance_start_datetime": "2020-09-02 00:00:00",  
43      "disturbance_end_datetime": "2020-09-03 10:24:00",  
44      "minimum_power_delivered": 0.018,  
45      "invulnerability_mean": 0.8,  
46      "invulnerability_std_dev": 0.4000000000000001,  
47      "recoverability_mean": 0.8165000328490883,  
48      "recoverability_std_dev": 0.3688498437001804,  
49      "resilience_mean": 0.8082500164245442,  
50      "resilience_std_dev": 0.3839434055427656  
51  },  
52 ▾  "resilience_data": [  
53 ▾  {  
62 ▾  {  
63      "minimum_power_load": -0.02,  
64      "minimum_power_delivered": 0.0,  
65      "disturbance_start_datetime": "2020-09-02 00:00:00",  
66      "disturbance_end_datetime": "2020-09-03 10:24:00",  
67      "invulnerability": 0.0,  
68      "recoverability": 0.16500032849088409,  
69      "resilience": 0.08250016424544204  
70  },  
71 ▾  {  
80 ▾  {  
89 ▾  {  
98 ▾  {  
107 ▾ {  
116 ▾ {  
125 ▾ {  
134 ▾ {  
143  ],  
144 ▾  "output_data": [  
176  }  
177  }
```

Figure 43. API Example of Executing a Microgrid Power Flow Simulation Series

## IV. ANALYSIS

### A. MODEL VALIDATION

The team sought to validate the efficiency and effectiveness of our novel resilience capability in the context of microgrid planning. Pedersen et al. (2000) posit design confidence can be obtained through utilization of the Validation Square (Figure 44). This method identifies qualitative and quantitative techniques to analyze approximate representations of systems. Utility and usefulness, rather than formal accuracy, drive the justification process. The guidance further stipulates theoretical and empirical methods to evaluate design structure and performance.

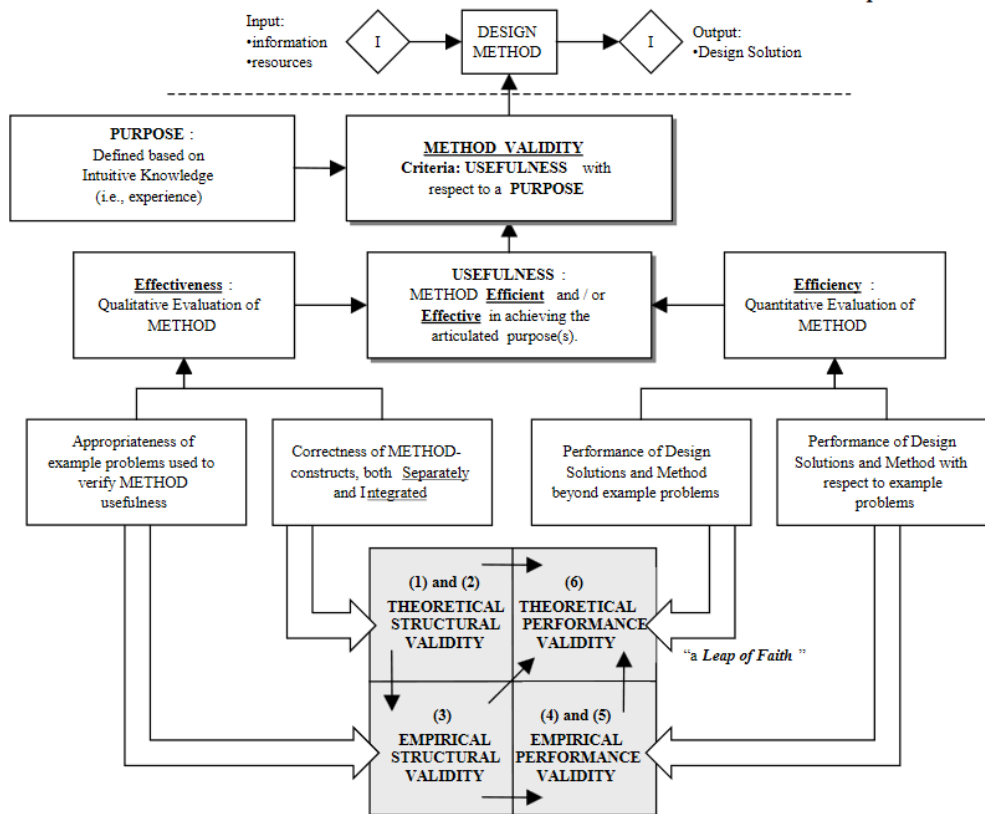


Figure 44. Design Method Validation. Source: Pedersen et al. (2000, 6).



Structurally, the resilience function is a mathematical description based on prior conceptualizations, and constrained by the interfaces and data structure within the existing MPT architecture. The tool’s foundational logic has withstood peer review and gained acceptance within the academic community (Anderson 2020; Giachetti et al. 2022; Reich and Oriti 2021). The work herein entails practical application of existing knowledge, as delineated in Chapter III. To further bolster “individual construct validity,” factor values were selected from trustworthy or representative sources. Energy resource attributes and performance parameters were well defined and often governed by industry standards. Power demand curves were sampled from active electrical microgrids. Resource failure probabilities warrant a caveat due to their imprecision. The assessment is prone to over- or under-estimation, injecting uncertainty into the analysis. However, the tool’s structure prioritizes code adaption and modification. As deterministic, distributed, and stochastic variables become better understood, the tool can be calibrated for more accurate predictions.

Performance evaluation proved more challenging based on resources available. Fundamentally, the tool evaluates microgrid performance under unlikely events. Disturbances, as rare events, introduce problems related to either limited data or ambiguous failure modes, thus real-world comparison is difficult. In lieu of experimental, empirical data, the tool was benchmarked against Anderson’s (2020) resilience model. Based on configurations and scenarios supported by both tools, a partial factorial DOE of 72 different sets of control variable values, was executed. Each DOE configuration was exercised through 100 independent simulation runs on each tool. The control variables, and the values they were each set to, are shown in Table 5. An analysis of variance (ANOVA) revealed model type produced a significant difference in resilience response, as indicated by a P-value less than  $\alpha$ , shown in Figure 45 (with a 95% confidence, and an  $\alpha$  value of 0.05, a P-value of 0.026 indicates statistical significance). Controlling for other identified factors, the plot of primary effects in Figure 46 generally agrees with the behaviors seen for MPT (Figure 51).

Table 5. MPT Resilience Metric Validation DOE Factors and Values

<u>Disturbance Type</u>	<u>Diesel Generator Quantity</u>	<u>Photovoltaic Generator Quantity</u>	<u>Battery Quantity</u>
Municipal Outage	1	1	1
Hurricane	2	5	5
Tsunami	3		
Earthquake			
Wildfire			
Cyber Attack			

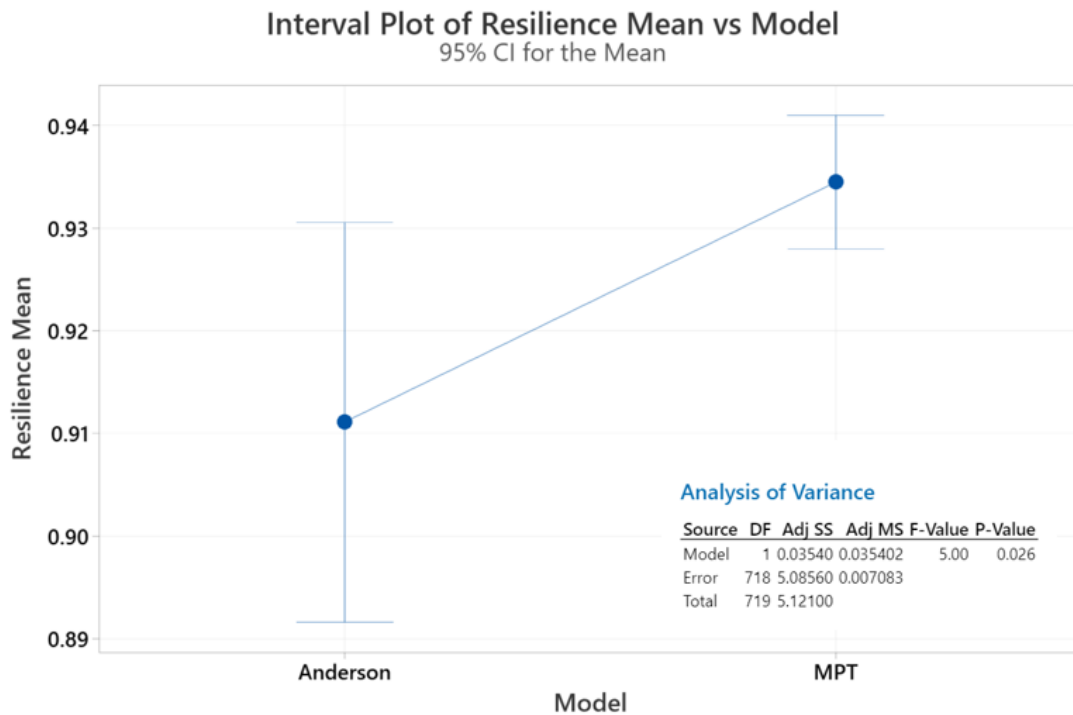


Figure 45. MPT and Anderson’s Cost Model ANOVA

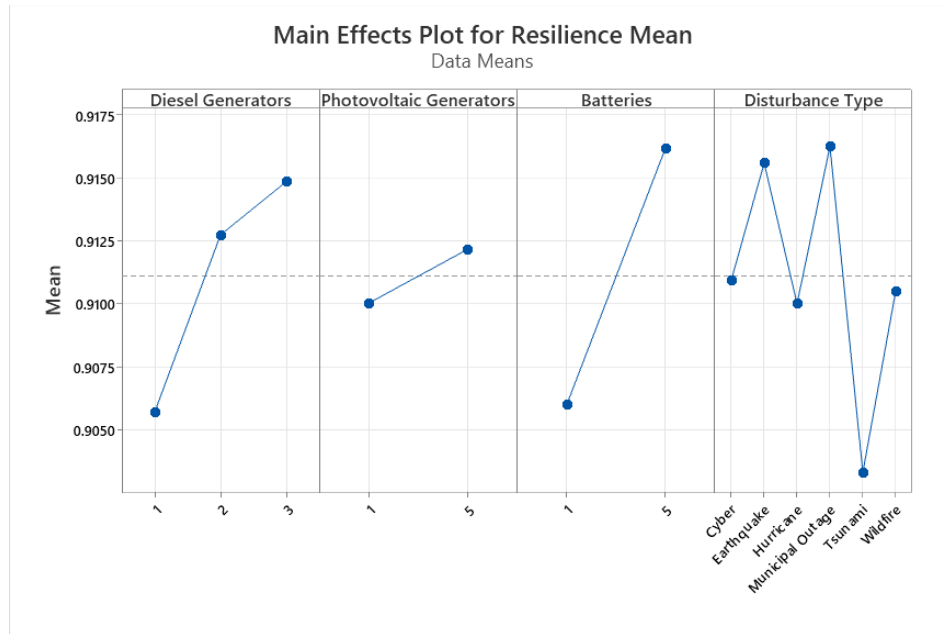


Figure 46. Resilience Main Effects for Anderson Model

Furthermore, factorial analysis of Anderson’s model indicated substantial differences between model logic and the resultant effects on resilience. For example, the effect of battery quantity was amplified while the effect of diesel generator quantity was somewhat dampened in comparison. Also, unexpected behavior was detected in certain specific interactions. For instance, a reduction (rather than increase) in resilience was seen when diesel generators were held to a quantity of three, and photovoltaic generator quantity was increased. As shown in the Pareto chart (Figure 47) and interaction plot (Figure 48), statistically significant factors were limited to batteries, two-way interactions between batteries and diesel generators, and diesel generators.

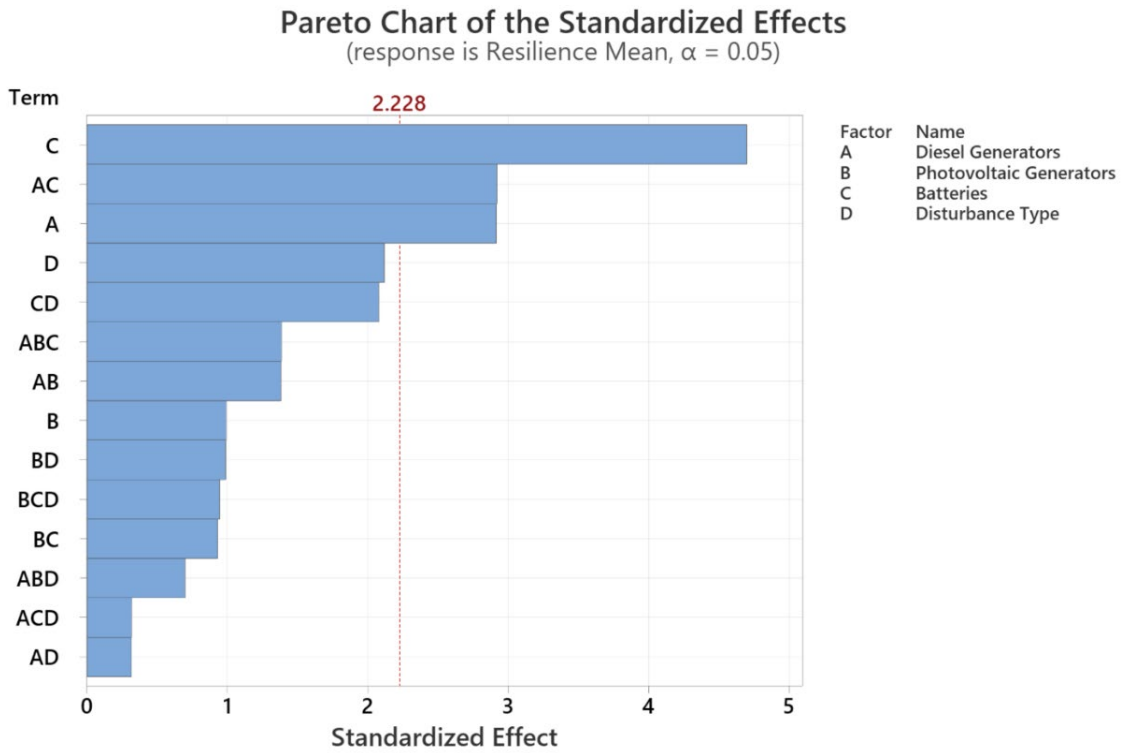


Figure 47. Resilience Pareto Chart for Anderson Model

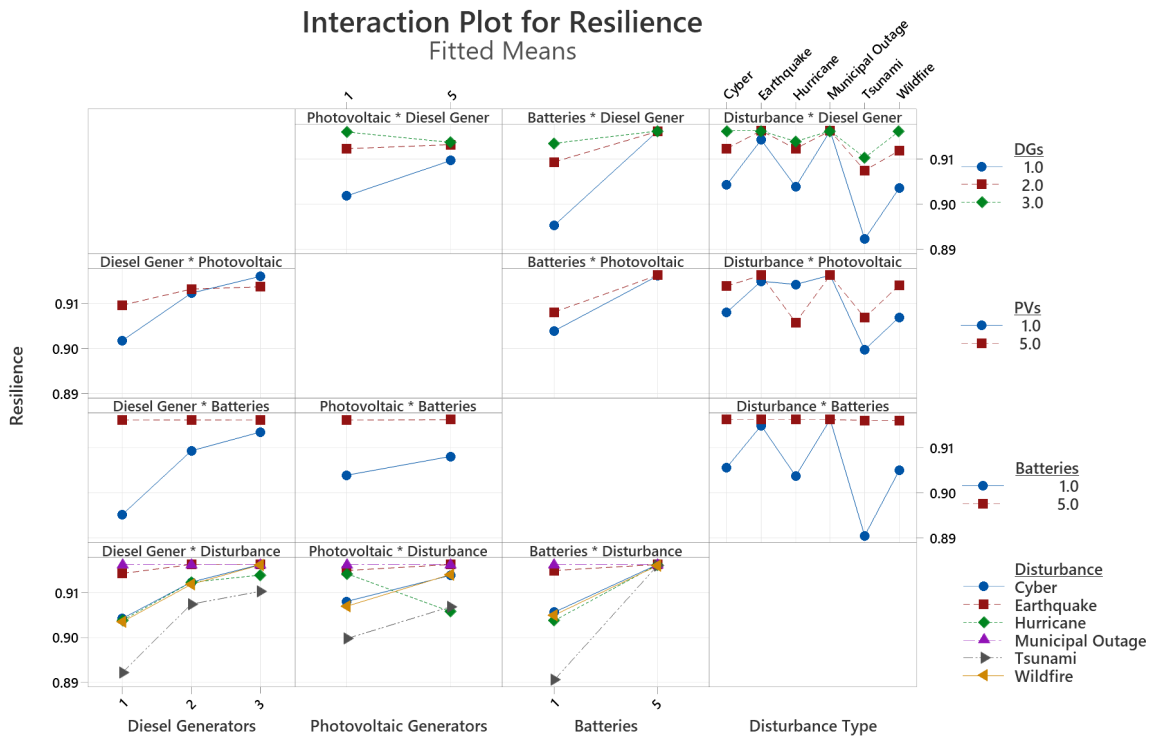


Figure 48. Resilience Interaction Plot for Anderson Model

Similar unintuitive findings were noted in prior validation efforts (Bolen et al. 2021). Speculatively, certain configurations may violate internal constraints or intended use cases. The team supposes differences in the application of conceptual and mathematical definitions for recovery, invulnerability, and resilience may account for a degree of variation. Collapsing continuous datasets into discrete metrics can lose meaning, or minimally comparative value, if not referenced identically. Moreover, complexities related to grid control and behavior would induce meaningful differences.

## B. EXPANDED CHARACTERIZATION

Broader MPT model characterization was pursued through further factorial analysis. Manipulated factors again included energy resources (diesel generators, photovoltaic generators, and batteries) and disturbances, as well as power profiles. The full list of controls and their values is provided in Table 6. The names of the power load profiles are effectively arbitrary; these power load profiles were sourced from the original MPT code base (all profiles beginning with “air\_terminal”) and Anderson’s model (“resilience\_cost\_201016”) (2020), and are representative of real-world installation microgrid power load data. Individual DER configuration parameters were consistent across replicates, as shown in Figure 49.

Table 6. MPT Resilience Metric Characterization DOE Factors and Values

<u>Power Load Profile</u>	<u>Disturbance Type</u>	<u>Diesel Generator Quantity</u>	<u>Photovoltaic Generator Quantity</u>	<u>Battery Quantity</u>
air_terminal	Municipal	1	1	1
air_terminal_3_anonymized	Outage	2	5	5
air_terminal_4_anonymized	Hurricane	3		
resilience_cost_201016	Tsunami			
	Earthquake			
	Wildfire			
	Cyber Attack			

```

# iterate through disturbances
for dist_param in dist_list:

    # iterate through DER quantities
    for dg_qty in dg_qty_list:

        for pv_qty in pv_qty_list:

            for b_qty in b_qty_list:

                run_name = "air_term_fact_DER_" + dist_param + "_dg" + str(dg_qty) + "_pv" + str(pv_qty) + "_b" + str(b_qty)

                query.params_as_dict({
                    "scenario_load": "air_terminal",
                    "num_runs": num_runs,
                    "sim_data_output": False,
                    "disturbance": dist_param,
                    "dg_qty": dg_qty,
                    "dg_power": 200,
                    "pv_qty": pv_qty,
                    "pv_power": 50,
                    "wt_qty": 0,
                    "wt_power": 150,
                    "b_qty": b_qty,
                    "b_discharge_power": 300.0,
                    "b_charge_power": 100.0,
                    "b_energy": 300.0,
                    "b_min_soc": 0.2,
                    "b_max_soc": .99,
                    "b_charge_eff": 0.95,
                    "b_discharge_eff": 0.95,
                    "b_charge_level": 300,
                })

```

Figure 49. DER Configuration Parameters

Disturbance types were categorical treatments ranging from cyber-attacks, earthquakes, hurricanes, municipal outages, tsunamis, and wildfires. Treatments were associated with specific failure mode likelihoods. Probabilities, as shown in Table 7, reflect multifaceted analysis informed by research, heuristics, and intuitions (Anderson 2020; Claire et al. 2019; Patel and Zaveri 2010). Analogous to resource pooling, for a given disturbance the resources failure probability was applied to each unique asset, not the entire class of DER. This design consideration allows the capture of improvements based on resource redundancy, not just improved energy capacity or generation.

Table 7. Disturbance Failure Rates

<b><u>Disturbance Type</u></b>	<b><u>DER Failure Rates</u></b>			
	<b>Diesel Generator</b>	<b>Photovoltaic Generator</b>	<b>Battery</b>	<b>Municipal Grid</b>
Cyber Attack	25%	25%	25%	100%
Municipal Outage	0%	0%	0%	100%
Hurricane	30%	70%	20%	100%
Earthquake	15%	25%	10%	100%
Tsunami	30%	50%	50%	100%
Wildfire	20%	40%	50%	100%

Of note, the tool forces the microgrid to operate in “island” mode (described in section I.A Background) during every iteration. The microgrid always meets demand for the selected power profile except when a disturbance degrades resources. The microgrids emphasized in this analysis were properly sized for their intended application. Power loads utilized were selected from the existing tool library, with an additional profile extrapolated from Anderson’s resilience cost model (2020). As shown in Figure 50, these power profiles provided significant differences to characterize (each color line represents a different profile; the x-axis is time, and the y-axis is power demanded). Each profile samples a fourteen-day period.

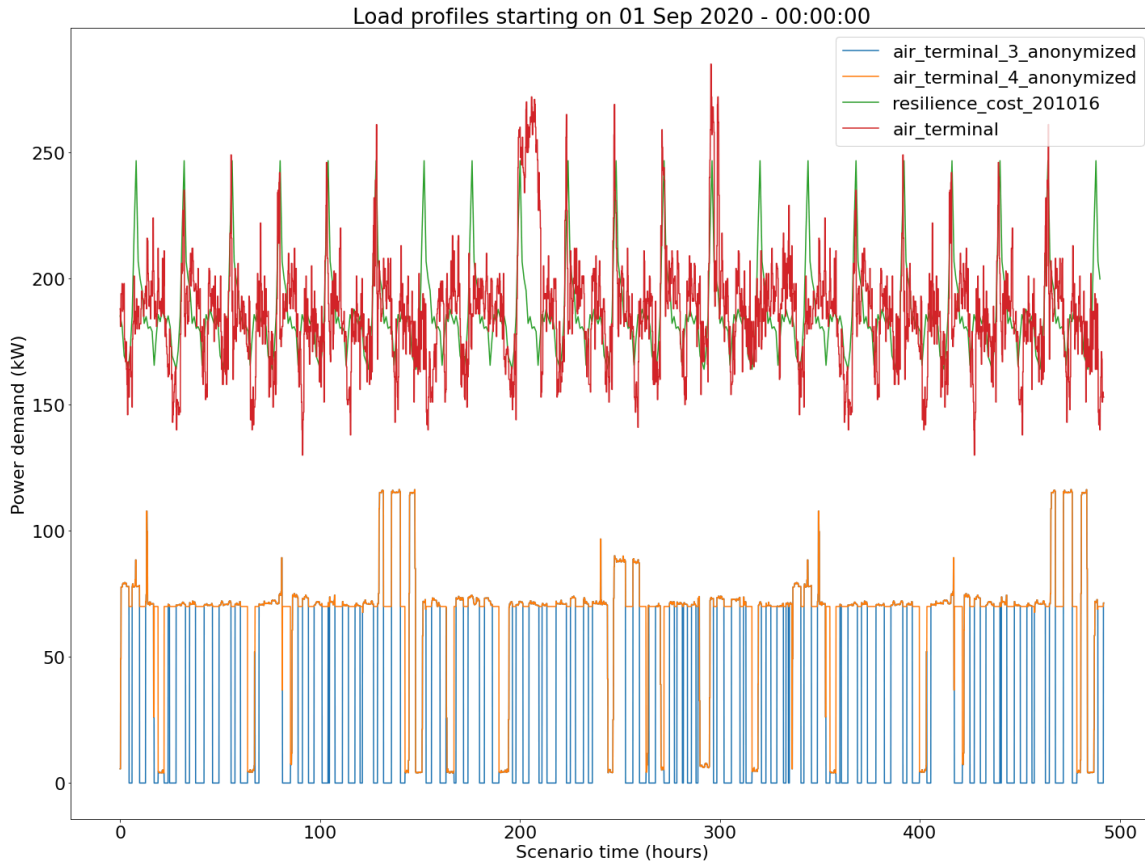


Figure 50. Analyzed Power Load Profiles

The design factors yielded 648 unique combinations. Each combination was simulated for one hundred replicates, generating 64,800 datasets. Responses gathered from each combination series were invulnerability, resilience, and recovery.

### C. CHARACTERIZATION RESPONSES

Factors were isolated to determine their respective independent effects on recovery, invulnerability, and resilience. The main effects plot (Figure 51) provides a graphical tool for quick, normalized comparisons between factors. As defined, resilience is a standard form linear equation dependent on equal parts recovery and invulnerability. Furthermore, since recovery and invulnerability exhibit comparable outcomes, observations are generalized.



Regarding energy resources the following are identified. First, the microgrid is highly sensitive to incrementing from a single diesel generator to a dual set. Interestingly, this sensitivity begins to dampen with the addition of a third unit. Second, enlarging the battery array capacity offers continuous, moderate improvement for quantities selected. Third, increasing the photovoltaic generator array size returns a flat, albeit slightly positive, response curve.

Disturbance type inspection revealed varied responses. Unsurprisingly, outcomes negatively trend with high failure rate scenarios that degrade high impact energy resources. Discussion continues during review of the generated Pareto plots.

Power demand profiles can be binned into two further categories, low and high demand. High demand includes anonymized data from the “air\_terminal” (no suffix) profile included with MPT, and Anderson’s resilience cost model load. Low demand consists of anonymized data from “air\_terminal\_3” and “air\_terminal\_4.” Notionally, demand may exhibit transient behavior, fluctuating based on seasonal changes and operational tempo. Clearly, high demand, relative to capacity, correlates with a decrease in recovery, invulnerability, and resilience metrics.

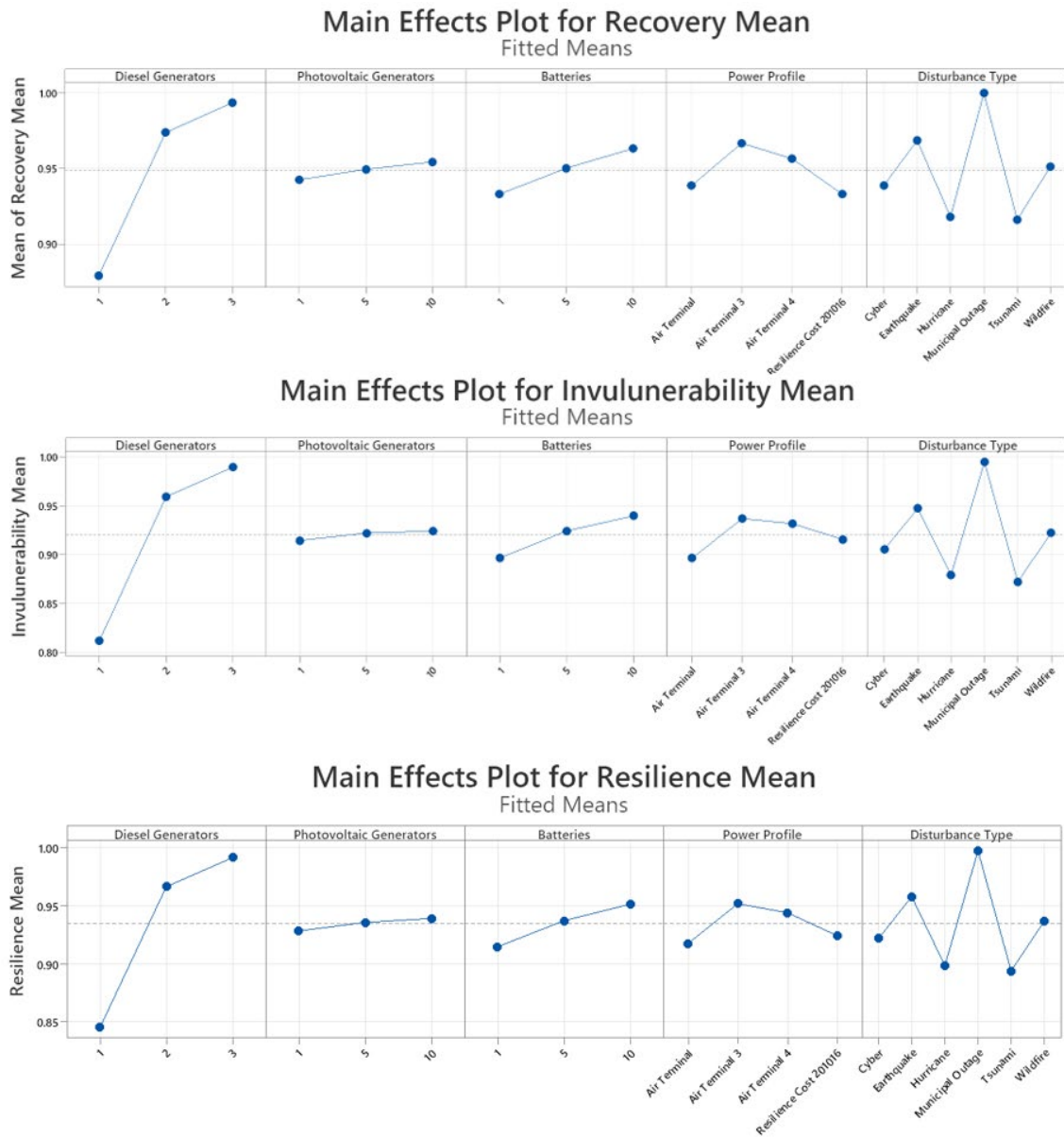


Figure 51. Resilience Metrics Main Effects

Standard effects are detailed in the Pareto plots shown in Figure 52, Figure 53, and Figure 54. With 95% confidence, the following statements hold true for the sample population. Diesel generators along with their two and three-way interactions, batteries, the interaction between batteries and disturbance type, and power profiles are the primary factors of influence. Additionally, photovoltaic generators and several combinations of two- and three-way interactions are significant. Order and magnitude of importance varies

between recovery, invulnerability, and resilience. For example, the interaction between disturbance type and power profile has a greater impact on recovery than invulnerability. Furthermore, recovery exhibits greater sensitivity to battery allocation than invulnerability.

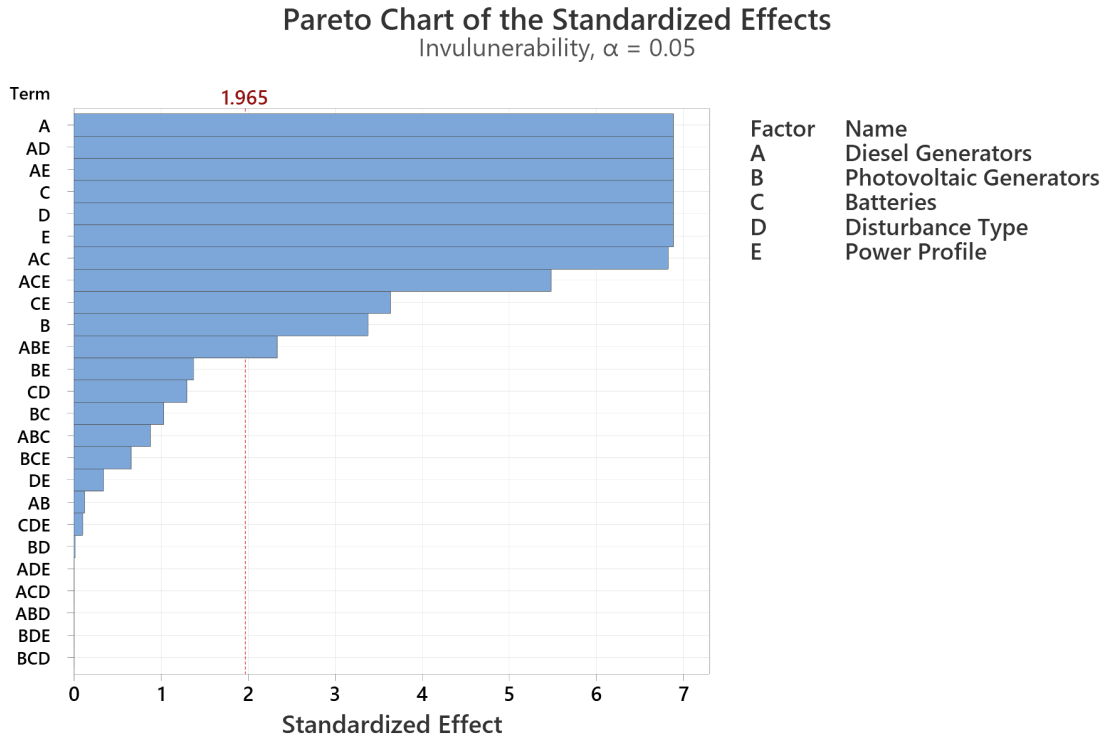


Figure 52. Invulnerability Effects Pareto Chart

### Pareto Chart of the Standardized Effects Recovery, $\alpha = 0.05$

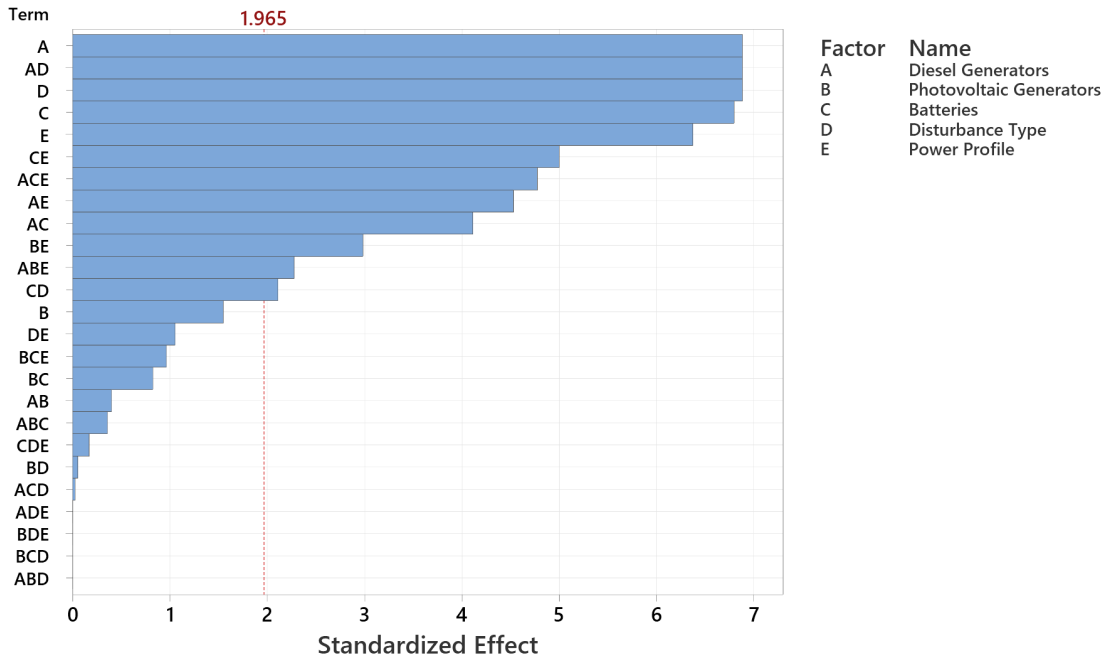


Figure 53. Recoverability Effects Pareto Chart

### Pareto Chart of the Standardized Effects Resilience, $\alpha = 0.05$

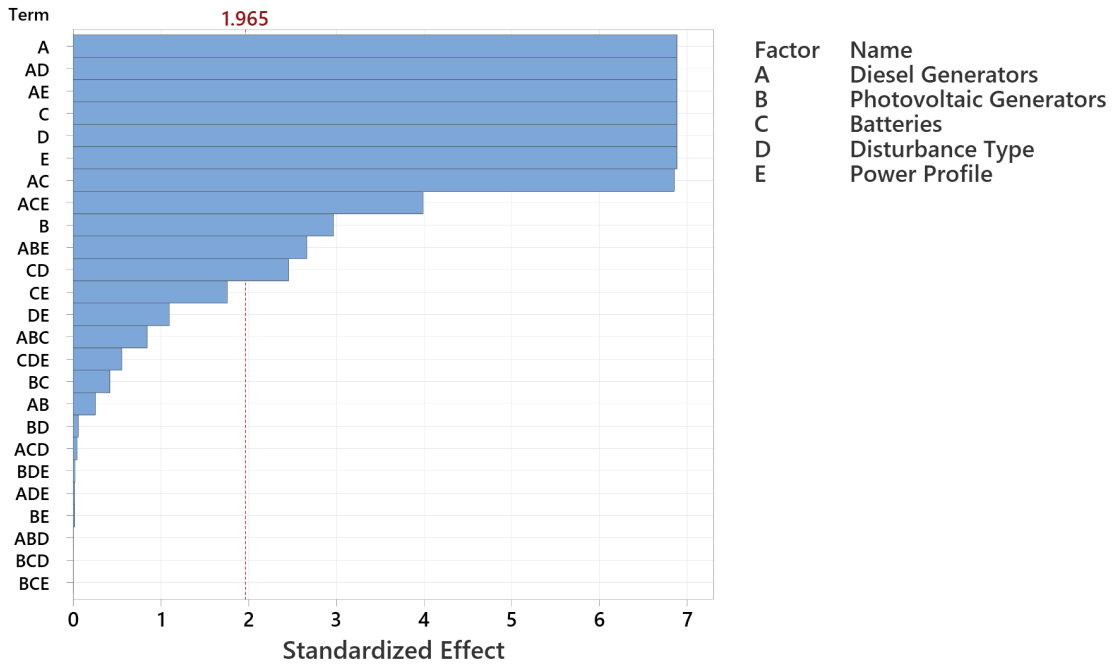


Figure 54. Resilience Effects Pareto Chart

Interaction plots are used to gain further insight. The plot matrix provides a method to interpret if two factors influence one another and therefore elicit a greater change in response than their individual contribution. Parallel response curves indicate no measured interaction between factors. Converging, or non-parallel, curves indicate an interaction occurs, where the magnitude of their respective slope differences indicates the strength of interaction. The resilience interaction plot (Figure 55) reaffirms relationships previously established. The weak connection between photovoltaics and resilience is observed. As shown, photovoltaic two-way interaction curves remain parallel for considered factors. Additionally, significant factors are confirmed. The resilience penalty associated with comparatively higher power demand loads and particularly punitive disturbances is substantially mitigated by redundant diesel generators.

# Interaction Plot for Resilience Mean Fitted Means

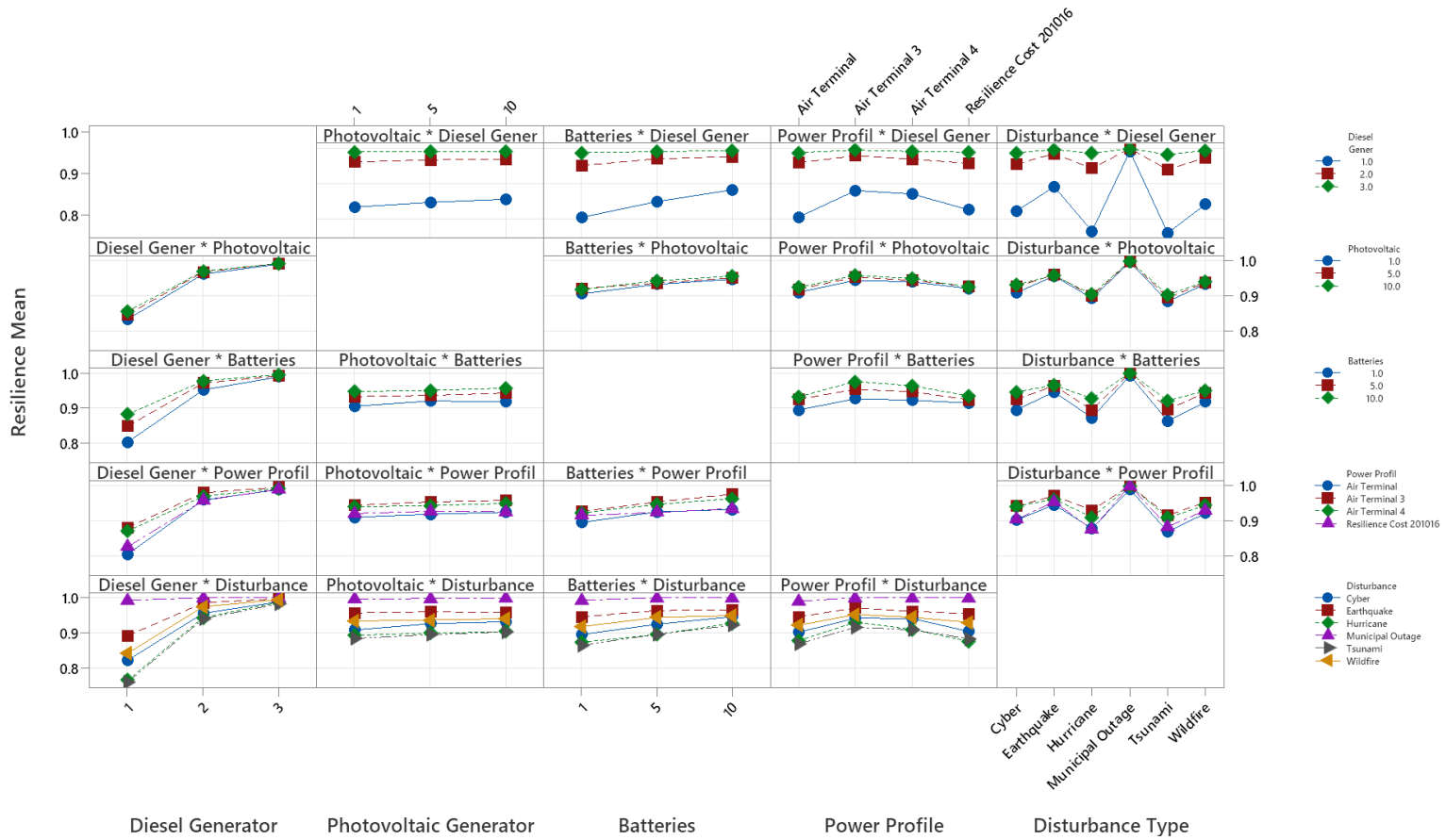


Figure 55. Resilience Metrics Effects Interaction Plot

THIS PAGE INTENTIONALLY LEFT BLANK

## V. CONCLUSIONS

In addition to providing a novel microgrid resilience analysis capability, the execution of this project addressed the research questions proposed in the Introduction section of this report. It also illuminated several tactics for efficiently and effectively performing software development in parallel, and proved the efficacy of implementing a tailored Agile workflow in lieu of a wholly traditional SE process. Finally, this effort identified objectives for follow-on and associated work also aimed at enhancing microgrid resilience.

### A. RESEARCH QUESTIONS

#### 1. Effective Integration of Tool Platform

The first of our research questions asked, “What is an effective way to integrate these microgrid resilience analysis methods and tools into a unified, efficient, and useful platform?” Leveraging the existing MPT software suite as a foundation, we were able to incorporate new functions and methods to provide microgrid resilience metrics calculation capability. These additions were structured such that they did not require significant modification of the existing software, and so that they would be accessible to future developers that wished to employ them in related operations and investigations. This permitted continued utilization of already-present MPT functionalities without negative or deleterious impact, while also enabling as-yet-conceived prospective functionalities with an expanded toolset. Thus, this work successfully demonstrated the effective combination of existing and new microgrid resilience analysis methods into a single software tool platform, while both maintaining and expanding its utility.

#### 2. Tool Platform Accessibility

Our next research question inquired, “How can intuitive, practical accessibility be provided for such a tool platform?” Again building upon foundational MPT capabilities, the team incorporated new API elements that adhered to established standards, facilitating access to the resilience analysis additions. This included adding the ability to query relevant



data constructs available within MPT, permitting users to utilize or modify already-defined microgrid and DER configurations. Alternatively, this also allowed the specification of entirely unique configurations. These API inclusions successfully enabled users at both “higher” administrative/planning and “lower” technical/engineering levels to access the resilience analysis enhancements via interfaces constructed around the existing MPT API. This granted both intuitiveness by avoiding the development of a disparate access methodology, and practicality by offering flexibility in the complexity of access requested.

### **3. Tool Platform Validity**

The final research question proposed was, “What measurements can be used to assess the validity of such a tool platform?” As the Analysis section explains, real-world microgrid disturbance events are sufficiently rare so as to preclude the correlation of actual to simulated data. However, the general trends in resilience in response to specific adjustments in microgrid architectures are well-examined and documented (Peterson et al. 2021; Giachetti et al. 2022; Anuat, Van Bossuyt, and Pollman 2021; Reich and Oriti 2021; Kain, Van Bossuyt, and Pollman 2021). In lieu of authentic data to compare against, the team instead constructed and executed a design of experiments, and demonstrated similar primary relationships between microgrid architecture composition and resilience value determinations. Further, we were also able to perform a congruent DOE on another, peer-reviewed microgrid modeling and resilience analysis tool (Anderson 2020), and again corroborated agreement between principal controlling variables and resilience value impacts. Hence, we successfully validated the MPT microgrid resilience analysis additions via comparison against behaviors articulated in literature, and those produced by another community-accepted microgrid model.

## **B. SOFTWARE DEVELOPMENT ARTIFACTS**

Software development involving multiple developers requires configuration control and management to establish task traceability and ensure successful collaboration. The configuration control management strategy was chosen early on to align closely with the source software development to enable upstream merging at the completion of the project.

## 1. Upstream Repository Merging

In an effort to adhere to robust software development principles, the team worked with the primary developer of MPT (Dr. Daniel Reich) to merge changes made over the course of this project upstream, in order to integrate the developed functionality enhancements. Since the original software development also continued in parallel, the team created a new branch and performed a Git merge against the upstream main MPT project. Many files included changes by developers from both projects, and manual inspection was required to combine modifications while avoiding conflicts. The resultant branch was then rigorously tested to ensure continued functionality.

Finally, the team created a merge request with the main MPT branch, and met with Dr. Reich to perform a line-by-line review and adjudication of our resilience calculation and API additions. Working together, the team and Dr. Reich created a series of squashed commits (GitLab n.d.), in order to group them together and keep the Git history tree concise. The final merge request UI and associated commentary are shown in Figure 56.

# Merge resilience analysis capstone efforts into microgrid

Edit Code

Merged Buetow, Thomas (CIV) requested to merge [fix/merge\\_from\\_resilience\\_cap...](#) into [master](#) Oct 7, 2022, 2:39 AM

Overview 1 Commits 22 Pipelines 0 Changes 11

Revert all previous merging, redo selective squashing of commits to preserve authorship. Additionally add load API updates with requested additions and update front-end to use new endpoint URI.

Approval is optional

Merged by Reich, Daniel (CIV) Oct 7, 2022, 1:07 PM Revert Cherry-pick

- Changes merged into `master` with [ce6372f9](#).
- Deleted the source branch.

0 thumbs up 0 thumbs down

Sort or filter

Buetow, Thomas (CIV) @thomas.buetow requested review from @daniel.reich Oct 7, 2022, 2:39 AM

Buetow, Thomas (CIV) @thomas.buetow assigned to @thomas.buetow Oct 7, 2022, 2:39 AM

Buetow, Thomas (CIV) @thomas.buetow · Oct 7, 2022, 2:40 AM  
TODO: make decision on `simulate_grid.py` changes. Author Developer

Buetow, Thomas (CIV) @thomas.buetow added 2 commits Oct 7, 2022, 1:03 PM

- [ddf3371d](#) - Correct api documentation and comments
- [d592b130](#) - Add resilience run script

[Compare with previous version](#)

Buetow, Thomas (CIV) @thomas.buetow marked this merge request as **ready** Oct 7, 2022, 1:03 PM

Reich, Daniel (CIV) @daniel.reich merged Oct 7, 2022, 1:07 PM

Figure 56. Upstream Merge Request with Original MPT Repository

## 2. Software and Operating Environment Requirements

The importance of configuration control of software versioning and managing dependencies was identified early within development. Developers within the team quickly discovered issues related to different versions of Python and associated libraries and packages, and quickly aligned to a common version to continue development throughout the project. To assist with versioning, two approaches were used to ensure future compatibility. First, the recommended version of Python is stated within the project

“README.md” file, and notes that other versions may be incompatible with this set of software. Second, within the code base exists a file called “requirements.txt” that can be invoked using Python’s integrated package installer, ensuring that all the Python package dependencies are installed with the correct versions before running any of the software. Instructions for this process are also included in the “README.md” file, and the specific software and package requirements for MPT are listed in Table 8. Additionally, due to the use of Jupyter notebooks for simplified interaction with the API and data visualization, an additional set of python software requirements are included within the “notebooks” directory of the Team Tom Capstone MPT code additions.

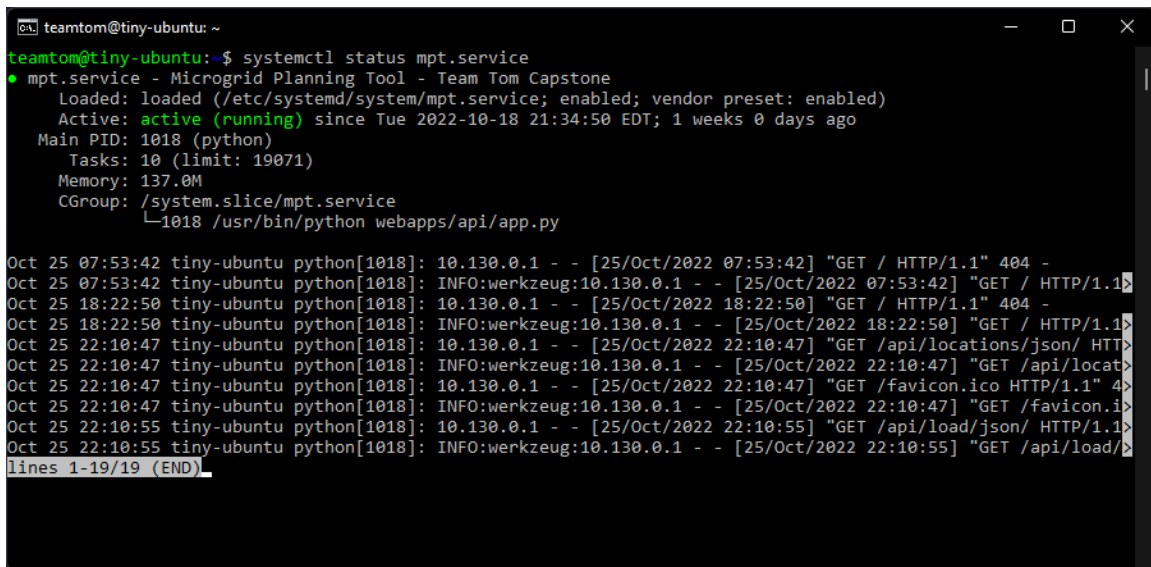
Table 8. List of MPT Software Dependencies and Versions

<u>Software/Package</u>	<u>Version</u>
Python	3.9
flask	2.1.0
flask-cors	3.0.10
flask-redoc	0.2.1
matplotlib	3.4.1
numpy	1.20.2
pandas	1.2.3
configargparse	1.5.2
flask-login	0.6.0
flask-session	0.4.0
flask-wtf	1.0.1
WTForms	3.0.1
bs4	0.0.1
requests	2.26.0

For development and testing, developers on the team used a common software program to make enhancements to the software. Each developer used Microsoft Visual Studio Code, which features integration with the Git protocol, to pull current branches, make changes, and commit changes back to the repository. This allowed for a common development environment and ensured an efficient collaboration solution for all developers.

### 3. Web Server Infrastructure

For the duration of this project, the infrastructure to run the API server consisted of an Ubuntu Linux virtual machine, maintained by the team. This allowed flexibility in software package management and ease of debugging during development, but was not suitable for long term hosting of the resilience APIs. Figure 57 shows a screen capture of the service status for this server while in use.



```
teamtom@tiny-ubuntu: ~
teamtom@tiny-ubuntu:~$ systemctl status mpt.service
● mpt.service - Microgrid Planning Tool - Team Tom Capstone
   Loaded: loaded (/etc/systemd/system/mpt.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-10-18 21:34:50 EDT; 1 weeks 0 days ago
     Main PID: 1018 (python)
        Tasks: 10 (limit: 19071)
       Memory: 137.0M
      CGroup: /system.slice/mpt.service
             └─1018 /usr/bin/python webapps/api/app.py

Oct 25 07:53:42 tiny-ubuntu python[1018]: 10.130.0.1 - - [25/Oct/2022 07:53:42] "GET / HTTP/1.1" 404 -
Oct 25 07:53:42 tiny-ubuntu python[1018]: INFO:werkzeug:10.130.0.1 - - [25/Oct/2022 07:53:42] "GET / HTTP/1.1" 404 -
Oct 25 18:22:50 tiny-ubuntu python[1018]: 10.130.0.1 - - [25/Oct/2022 18:22:50] "GET / HTTP/1.1" 404 -
Oct 25 18:22:50 tiny-ubuntu python[1018]: INFO:werkzeug:10.130.0.1 - - [25/Oct/2022 18:22:50] "GET / HTTP/1.1" 404 -
Oct 25 22:10:47 tiny-ubuntu python[1018]: 10.130.0.1 - - [25/Oct/2022 22:10:47] "GET /api/locations/json/ HTTP/1.1" 200 -
Oct 25 22:10:47 tiny-ubuntu python[1018]: INFO:werkzeug:10.130.0.1 - - [25/Oct/2022 22:10:47] "GET /api/locat>
Oct 25 22:10:47 tiny-ubuntu python[1018]: 10.130.0.1 - - [25/Oct/2022 22:10:47] "GET /favicon.ico HTTP/1.1" 404 -
Oct 25 22:10:47 tiny-ubuntu python[1018]: INFO:werkzeug:10.130.0.1 - - [25/Oct/2022 22:10:47] "GET /favicon.i>
Oct 25 22:10:55 tiny-ubuntu python[1018]: 10.130.0.1 - - [25/Oct/2022 22:10:55] "GET /api/load/json/ HTTP/1.1" 200 -
Oct 25 22:10:55 tiny-ubuntu python[1018]: INFO:werkzeug:10.130.0.1 - - [25/Oct/2022 22:10:55] "GET /api/load/>
lines 1-19/19 (END)
```

Figure 57. MPT Development Server Status Information Sample

As a result of the upstream code merging described in the Upstream Repository Merging section, this project is now hosted on NPS servers (<https://microgrid.nsetti.nps.edu>). However, due to NPS infrastructure security requirements, the resilience APIs will time out after approximately 60 seconds of wait time, preventing the execution of lengthy simulation series consisting of many individual simulation runs. A discussion of proposed technical improvements that may solve this problem follows, however, until such a solution is implemented, it is recommended to run the software locally if possible.

### **C. HYBRID AGILE METHOD**

As part of the overall SE process, the team used a hybrid approach to the Agile method. This was realized as an initial use of the more traditional waterfall process, followed by a transition to Agile to better accommodate team resources while continuing to meet requirements. Initial stakeholder analysis, needs identification, and requirements analysis was performed using the waterfall process, and software development, implementation, and analysis followed the Agile methodology. This hybrid strategy allowed the team to tailor the SE process to work with improved efficiency within the constraints of the project.

Ultimately, 13 sprints were completed, with 11 sprints dedicated to project and report development. The final two sprints were dedicated to project reviews and composition of a journal manuscript in order to publish project findings and accomplishments. Each sprint planning session allowed the team to develop focused tasking for individual members. This allowed establishment and execution of more realistic objectives, and delivery of improved products at sprint conclusions. At the end of each sprint, with the participation of working level advisors, a team retrospective was held and documented in an engineering notebook. These retrospectives reviewed work completed, compared against sprint objectives, and identified goals for the next sprint. An example retrospective and planning engineering notebook entry is provided in Figure 60 in Appendix D.

### **D. RECOMMENDATIONS FOR FUTURE WORK**

Throughout project development, several areas were identified for future work. These areas were split between technical improvements and deferred scope. Within technical improvements, resilience modeling and the MPT suite were identified as areas that could benefit from additional work in the future. Due to limited time and a focus on leveraging existing team member skill sets to maximize completion of deliverables, several stakeholder needs and requirements were deferred. Requirements traceability enabled a targeted strategy for implementation throughout the project.

## 1. Technical Improvements to Resilience Modeling

Resilience modeling depends on determining other criteria such as invulnerability and recoverability. Under the current definition of invulnerability, it is represented by the ratio of singular values of power delivered and power demanded. However, both of these values are time-varying, and an accepted standard method for reducing sets of power demanded and power delivered data to single respective values has not yet been established. To grant consistency across all implementations of the invulnerability calculation (and consequently, resilience modeling and analysis)—pertinently including instances when multiple distinct disruptions may be in effect—it would be beneficial to have available a clearly-defined, rational, and meaningful method for establishing singular power demanded and power delivered values.

Disruption types and effects play a central role in the analysis of microgrid resilience. As noted previously, several types of disruptions may occur to any type of microgrid, but each disruption may have a different probability of occurrence. Expanding resilience metrics to include probabilities for local disruptions could improve the fidelity of resilience modeling for specific microgrid architectures. It should be noted that alternative resilience metrics, for example the expected electrical disruption mission impact (EEDMI), do incorporate disruption probabilities into their calculations (Peterson et al. 2021).

Presently, the available power load profiles for performing simulations are limited to data sets representative of a few specific locations. Microgrid performance is important to multiple DOD installations around the world, and additional data sets related to different geographical installations would be beneficial for further determining factors that affect microgrid performance. For example, the solar and wind turbine power available may be significantly different in Guam than it is in Rota, Spain. Additionally, seasonal variations in specific locations may affect microgrid design and performance. Establishing standard practices for providing sufficiently inclusive data sets for microgrid power loads and generative capacity at disparate global locations, including an adequate quantity of seasonal changes and impacts, would significantly improve the overall quality of analysis of resilience modeling.

In general, resilience modeling is important for engineering and technical personnel responsible for design decisions for a particular site. What the resilience modeling does not tell the end user is impact to the mission. If a commanding officer or military program manager is not skilled with microgrids, they may not understand the mission impact of certain levels of resilience for a particular microgrid. The ability to quantify mission impact based on resilience level and specific microgrid mission would support critical decision making for high level staff. Again referencing an alternative metric, mission dependency index (MDI) does include such mission-specific considerations in its value determination (Grussing et al. 2010), and this in turn is used within the EEDMI calculation (Peterson et al. 2021). However there exist subjectivity and application consistency concerns with the utilization of MDI (Kujawski and Miller 2009).

A potential method to help quantify mission impact would be to establish a standard for “levels of acceptability” for resilience modeling. The levels of acceptability would provide system owners or program managers with performance thresholds to target during the design or upgrade of a microgrid architecture. For example, a stipulation that a given microgrid architecture should meet a minimum 0.97 resilience score with maximum 0.02 standard deviation over a well-defined set of simulations could be enforced as a requirement for permanent field installation of that architecture. Similarly, a relaxed condition of meeting a minimum simulated 0.85 resilience score with maximum 0.05 standard deviation could gate temporary field installation of a given microgrid architecture. Incorporation of such standards would help guide application of resilience metrics, and establish their impacts to the mission.

## **2. Technical Improvements to MPT Suite**

The MPT suite includes a wide range of features that enable various fields of analysis for planning microgrid architectures. However, throughout the incorporation of additional features into the suite, several areas were identified for technical improvement that would provide the end user with additional capabilities. Primarily, these areas involved incorporating additional features to support more complex microgrid architectures and



disturbance scenario conditions. Additionally, the ability to graphically interact with resilience modeling tools was identified for future improvements.

The ability to define and control disruptions within scenarios would increase the fidelity of the microgrid simulations. Currently, disruptions occur at a single time within the scenario defined as the “disturbance start time.” However, being able to define different time points in the scenario for a disruption to occur would enable expanded analysis capabilities within the MPT suite. Additionally, the ability to create and control multiple disturbance events within a single simulation scenario would provide further broadened levels of modeling capabilities to the suite.

Incorporating the ability to simulate non-trivial microgrid architectures would enable more complex resilience analyses. Currently, resilience analysis is defined for standard “single node” microgrid configurations, wherein all DERs are centrally connected to one another. Expanding this to include non-trivial microgrid architectures or multi-nodal microgrids would significantly enhance the MPT suite’s capability for complex resilience analysis. Within these microgrid architectures, the ability to define how disruption of one node of a microgrid could disconnect other DERs (generators and loads) from the rest of the microgrid would have significant effects on overall resilience. Similarly, the ability to define load shedding, where non-critical loads are identified and indicated to be disconnected first when power generation capacity is no longer sufficient to meet overall power demand, would enable yet another level of more realistic resilience analysis.

Providing the capability to define detailed specifications for microgrid loads in a microgrid configuration would further allow for expanded microgrid resilience analysis. MPT suite capabilities currently only allow for the definition of a single aggregate power demand, which simplifies the overall analysis, but prevents the utilization of the MDI and EEDMI metrics, as well as the implementation of load shedding logic. In addition to allowing for microgrid load decomposition and description, the ability to define reliability information for microgrid components, such as mean time between failures (MTBF), would also enhance the potential analysis capabilities for the resilience modules within MPT.

Outside of the configurations of microgrid architectures, the MPT suite is not yet capable of running large simulations from a web-based interface. To get sufficient data for stochastic modeling, the resilience simulations need many sets of multi-simulation runs performed. Under the current software architecture, this process could take many hours to execute, and was only possible using developmental tools on an isolated server environment. This is because Python threading does not allow work to be spread across multiple processors by default, as shown in Figure 58. To resolve this issue, the software’s “Simulation” class could be modified to utilize Python’s multiprocessing capabilities to perform simultaneous simulations for Monte Carlo analysis. Python’s multiprocessing capabilities do not include the ability to pass data between simulation objects, and would require additional re-architecting for portions of the simulation class. Alternatively, a queuing/token system could be developed, and would allow for end users to schedule large simulation runs in the background, and would notify them when their simulations have completed and results are available.

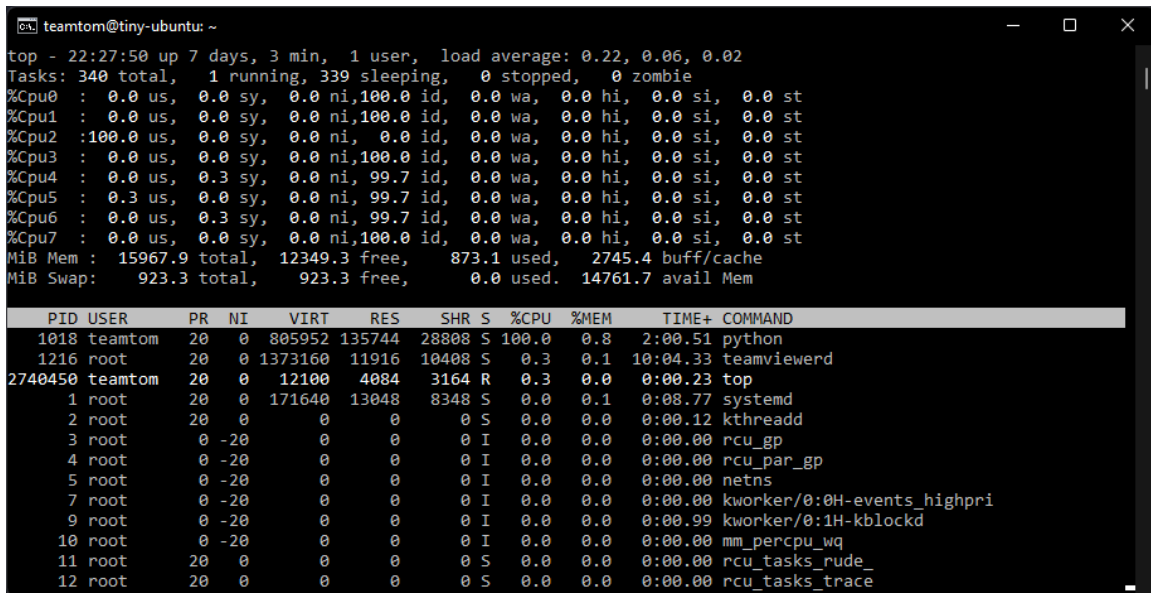


Figure 58. CPU Usage During Microgrid Simulation

Originally, the team planned to incorporate a web-based user interface for full access to the resilience modeling tools, similar to how they exist for the simulation and

rightsizing modules. However, due to limited development time and lack of existing technical skill set within the team, this was deferred. As an alternative, a Jupyter notebook paradigm was developed to enable simplified interaction with the resilience modeling APIs. The notebook capabilities allow for straightforward interaction with the API with graphical responses, while alleviating the requirement to develop a web-based front end for user interaction. Figure 59 shows the example notebook developed to interact with the various API endpoints.

```
Resilience_analysis.ipynb
lience Notebook Demo > M*Review the available DER generators > M*Select a DER scenario to further investigate > ## Narrow down on details of one generator scenario ##
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Variables Outline ... Python 3.10.8 64-bit

Review the available DER generators

This is part of use case Create DER Models

### Get list of grid generators from server ###
query.api_endpoint("/api/gridmakeup/json")
query.params_as_string("")

gen_data = query.fetch_json_data()

for k,v in gen_data["data"].items():
    print(k)

[5] ✓ 0.3s Python

...
resilience_cost_201108
air_terminal_2c
air_terminal_3_quadruple
resilience_cost_201016
air_terminal_3a
air_terminal_3
air_terminal_1
air_terminal_2b
air_terminal_2

Select a DER scenario to further investigate

### Narrow down on details of one generator scenario ###
gen_scenario = "air_terminal_3"

for l in gen_data["data"][gen_scenario]:
    if l["type"] == "Battery":
        print(l["id"], l["type"], l["attributes"]["power_rating"], "kW", l["attributes"]["energy_rating"], "kWh")
    else:
        print(l["id"], l["type"], l["attributes"]["power_rating"], "kW")

[6] ✓ 0.3s Python

...
pv1 SolarPhotovoltaicPanel 50.0 kW
dg1 DieselGenerator 200.0 kW
b1 Battery 300.0 kW 300.0 kWh
```

Figure 59. API Endpoint Interaction Example Jupyter Notebook

### **3. Deferred Stakeholder Needs**

During the requirements identification phase, stakeholder needs were collected during outreach and collaboration with sponsors, advisors, and stakeholders. As the project was tailored based on customer interest, several needs were tagged as deferred to enable traceability and to focus project development. Through this traceability, the deferred stakeholder needs can easily be identified for future work. Table 2 in section II.C Requirements Identification lists all identified stakeholder needs, including those with “deferred” tags. Overall, 39 of the 65 defined system requirements were implemented by the project team, leaving 29 for future efforts to address.

### **4. Deferred System Requirements**

Similar to stakeholder needs, system requirements that were developed early during the requirements identification phase were refined and tagged with either a “select” or “defer” notation, as discussed in section II.C Requirements Identification and shown in Table 9 in the Appendix A. These deferred requirements can also be easily identified for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX A. SYSTEM REQUIREMENTS

Table 9. System Requirements

<u>Number</u>	<u>Name</u>	<u>Description</u>	<u>Labels</u>
SRQ1	Basic		
SRQ1.1	Improve DOD Energy Security	The Microgrid Planning Tool (MPT) resilience module optimizations shall show an aggregate improvement over traditional methodologies.	Select (Threshold)
SRQ1.1.1	Decision Aid	The MPT module shall provide resiliency metrics to aid decision making.	Select (Threshold)
SRQ1.1.2	Recoverability	The MPT module shall provide recoverability analysis metrics for microgrids.	Select (Threshold)
SRQ1.1.3	Invulnerability	The MPT module shall provide invulnerability analysis metrics for microgrids.	Select (Threshold)
SRQ1.2	MSOSA Digital Twin Model	A digital twin system shall be provided by utilizing the MSOSA system modeling software suite.	Defer
SRQ1.2.1	Digital Model Verification	The digital model shall be evaluated against verified experimental data and alternate models.	Defer
SRQ1.2.2	Digital Model Integration	The digital model shall have a defined interface control document to support integration with the MPT.	Defer
SRQ1.2.3	Disturbance Module	The digital model disturbance module shall include multiple use cases and durations.	Defer
SRQ1.2.4	Resilience Module	The digital model shall be capable of performing a user selected resilience analysis on the selected model.	Defer
SRQ1.2.5	Model Source Data	The digital model libraries shall be accurate and continuously maintained.	Defer
SRQ1.2.5.1	Component Library	The digital model component library shall be accurate and continuously maintained.	Defer
SRQ1.2.5.2	DER Library	The digital model DER library shall be accurate and continuously maintained.	Defer
SRQ1.2.5.3	Battery Library	The digital model battery discharge / charge cycle library shall be accurate and continuously maintained.	Defer
SRQ1.2.6	Control Module	The digital model shall utilize a state machine for the control module.	Defer
SRQ1.3	Microgrid Planning Tool Resilience Module	The resilience module shall integrate with the existing MPT baseline.	Select (Threshold)
SRQ1.3.1	Disturbance Events	The resilience module shall incorporate global and local disturbance events.	Select (Objective)

<b><u>Number</u></b>	<b><u>Name</u></b>	<b><u>Description</u></b>	<b><u>Labels</u></b>
SRQ1.3.2	Resilience Analysis	The resilience module shall aid resilience analysis.	Select (Threshold)
SRQ1.3.3	Defined Interface	The resilience module shall have a defined interface control document to support integration with a digital model.	Select (Threshold)
SRQ1.3.4	Source Data	The libraries access by the resilience module shall be accurate and continuously maintained.	Defer
SRQ1.3.5	Legacy Support	The resilience module shall support rightsizing functionality currently available in the NSETTI baseline.	Select (Objective)
SRQ1.4	Data Distribution	An application, process, or method shall be created to share data between the digital model and MPT functions.	Defer
SRQ1.4.1	Interface UI	Developed application interfaces shall provide an UI to aid data transfer.	Defer
SRQ1.4.2	Data Transfer Mechanism	The system shall provide a standardized method for transferring data from the digital model to the MPT resilience module.	Defer
SRQ1.5	User Customization	The system shall allow the user to customize model data.	Select (Objective)
SRQ1.5.1	Predefined Power Profiles	The user shall be able to select predefined system power profiles.	Select (Threshold)
SRQ1.5.2	Custom Power Profiles	The user shall be able to select critical and non-critical load profiles.	Defer
SRQ1.5.3	Load Shedding Profiles	The user be able to select a predefined load shedding profile.	Defer
SRQ1.5.4	User Profiles	The resilience module shall allow user to create secure individual profiles.	Defer
SRQ1.5.5	Database	The resilience module shall utilize databases for persistent data storage.	Defer
SRQ1.6	External Geo Location Environment Services	The resilience module shall integrate existing location data.	Defer
SRQ1.6.1	Solar Irradiance Data	The resilience module shall integrate existing location-based irradiance data.	Defer
SRQ1.6.2	Wind Data	The resilience module shall integrate existing location-based wind data.	Defer
SRQ1.6.3	Climate Data	The resilience module shall integrate existing location-based climate data.	Defer
SRQ1.6.4	Consumption Data	The resilience module shall integrate existing location-based, climate dependent energy consumption data.	Defer
SRQ2	Interoperability		

<b><u>Number</u></b>	<b><u>Name</u></b>	<b><u>Description</u></b>	<b><u>Labels</u></b>
SRQ2.1	Free and Open Standards	Software functionality shall use free and open-source tools where possible.	Select (Threshold)
SRQ2.1.1	Software License	The interactive UI shall use open or freely available tools to generate and display the UI elements and interactive features.	Select (Threshold)
SRQ2.1.2	Reuse	Software shall provide modular and modifiable functions to support ease of integration and code reuse for future development efforts.	Select (Threshold)
SRQ2.1.3	Legacy	Software shall support legacy software and protocols where possible.	Select (Threshold)
SRQ2.2	Published APIs	The system shall provide a documented list of APIs and how to interact with the system.	Select (Threshold)
SRQ3	Suitability Requirements		
SRQ3.1	Design	This is an unconstrained developmental design. The number of features is bounded by our technical ability and schedule.	Select (Threshold)
SRQ3.1.1	Name Conventions	The system shall use appropriate, consistent naming conventions.	Select (Threshold)
SRQ3.1.2	Modification History	The system shall provide detailed release notes summarizing code changes.	Select (Threshold)
SRQ3.1.3	Nesting	New code shall be limited to three levels of nesting or less.	Select (Threshold)
SRQ3.1.4	Module Description	Software modules shall provide function description including dependent variables.	Select (Threshold)
SRQ3.1.5	Efficient Code	Code shall utilize limited computer resourcing.	Select (Threshold)
SRQ3.1.6	Readability	Code shall adhere to consistent indentation style, minimize variable length, limit line length, and minimize module length.	Select (Threshold)
SRQ3.1.7	Code Grouping	Discrete tasks shall be contained in separate code blocks.	Select (Threshold)
SRQ3.1.8	Application Permissions	The system shall limit access to application elements based on user privileges.	Select (Threshold)
SRQ3.2	Reliability		
SRQ3.2.1	System Resource Reliability	The Interactive UI shall provide model data and simulation results when requested by the user 99% of the time when the system is normally available.	Select (Threshold)
SRQ3.3	Availability		



<b><u>Number</u></b>	<b><u>Name</u></b>	<b><u>Description</u></b>	<b><u>Labels</u></b>
SRQ3.3.1	Interactive UI Uptime	The Interactive UI shall be available for use and functional 95% of the time (threshold) and 99% of the time (objective), averaged over 30 days.	Select (Threshold)
SRQ3.4	Maintainability		
SRQ3.4.1	Software Authoring Best Practices	Software writing shall follow best practices for structure and documentation of code.	Select (Threshold)
SRQ3.4.2	Software Development Best Practices	Versioning tools shall be used to provide commit history and rationale during development of code.	Select (Threshold)
SRQ3.4.3	System User Manual	A system user manual shall be developed to document digital model function and capability.	Defer
SRQ3.5	Survivability		
SRQ3.5.1	Software Runtime Environment Security	Software shall run on a host system that is kept up to date with security patches, following industry best practices.	Select (Objective)
SRQ3.5.2	Software Reliability	Software backups shall be taken to prevent accidental data loss using a continuous delta backup solution or equivalent.	Select (Objective)
SRQ3.6	Personnel, Safety, Human Factors, and Environmental Considerations		
SRQ3.6.1	Accessibility	Software UI shall conform to Section 508 standard accessibility requirements where applicable.	Select (Objective)
SRQ3.6.2	Operator feedback	The UI shall inform the operator that input requests are being processed following industry best practices.	Select (Threshold)
SRQ3.6.4	Software Response	Software inputs shall consistently produce perceptible response outputs.	Select (Threshold)
SRQ3.6.5	Data Entry	The software shall provide a positive feedback to the user of the acceptance or rejection of a data entry.	Select (Threshold)
SRQ3.6.6	Abbreviations	When abbreviations, mnemonics, or codes are used to shorten data entry, they shall be distinctive and have an intuitive relationship or association to normal language or specific job-related terminology.	Select (Threshold)
SRQ3.6.7	Standardized Content Display	The content of displays within a system shall be presented in a consistent, standardized manner.	Select (Threshold)
SRQ3.7.3	Input Complexity	Software shall limit input complexity where applicable.	Select (Threshold)
SRQ4	Key Performance Parameters		

<u>Number</u>	<u>Name</u>	<u>Description</u>	<u>Labels</u>
SRQ4.1	KPP1	The resilience module shall provide microgrid resilience analysis for simulations executed within the MPT tool.	Select (Threshold)
SRQ4.2	KPP2	The resilience module shall perform microgrid resilience analysis for well-defined MSOSA microgrid architectures.	Defer
SRQ4.3	KPP3	The resilience module shall translate well-defined MSOSA microgrid architectures into design parameters parsable by the MPT tool.	Defer

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX B. STAKEHOLDER NEEDS TO SYSTEM REQUIREMENTS TRACEABILITY

Table 10. Stakeholder Needs to System Requirements Traceability

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	NAME	NUMBER	STATUS	DESCRIPTION
SN1.1 - Primary				
SN1.1.1 - Improve DOD energy security analysis techniques.	KPP3	SRQ4.3	Defer	The resilience module shall translate well-defined MSOSA microgrid architectures into design parameters parsable by the MPT tool.
	KPP2	SRQ4.2	Defer	The resilience module shall perform microgrid resilience analysis for well-defined MSOSA microgrid architectures.
	KPP1	SRQ4.1	Select (Threshold)	The resilience module shall provide microgrid resilience analysis for simulations executed within the MPT tool.
	Improve DOD Energy Security	SRQ1.1	Select (Threshold)	The Microgrid Planning Tool (MPT) resilience module optimizations shall show an aggregate improvement over traditional methodologies.
SN1.1.2 - Evaluate resilience, recoverability, and invulnerability of microgrids.	Recoverability	SRQ1.1.2	Select (Threshold)	The MPT module shall provide recoverability analysis metrics for microgrids.
	Improve DOD Energy Security	SRQ1.1	Select (Threshold)	The Microgrid Planning Tool (MPT) resilience module optimizations shall show an aggregate improvement over traditional methodologies.
	Invulnerability	SRQ1.1.3	Select (Threshold)	The MPT module shall provide invulnerability analysis metrics for microgrids.
	Decision Aid	SRQ1.1.1	Select (Threshold)	The MPT module shall provide resiliency metrics to aid decision making.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
SN1.1.3 - Advance toolkit to aid decision makers in microgrid design architecture.	Improve DOD Energy Security	SRQ1.1	Select (Threshold)	The Microgrid Planning Tool (MPT) resilience module optimizations shall show an aggregate improvement over traditional methodologies.
	Decision Aid	SRQ1.1.1	Select (Threshold)	The MPT module shall provide resiliency metrics to aid decision making.
SN1.1.4 - Provide interface tool to integrate digital model and web tool.	Software License	SRQ2.1.1	Select (Threshold)	The interactive UI shall use open or freely available tools to generate and display the UI elements and interactive features.
	System User Manual	SRQ3.4.3	Defer	A system user manual shall be developed to document digital model function and capability.
	Application Permissions	SRQ3.1.8	Select (Threshold)	The system shall limit access to application elements based on user privileges.
	Data Entry	SRQ3.6.4	Select (Threshold)	The software shall provide active feedback to the user of the acceptance or rejection of a data entry.
	Software Response	SRQ3.6.3	Select (Threshold)	Software inputs shall consistently produce perceptible response outputs.
	Input Complexity	SRQ3.6.7	Select (Threshold)	Software shall limit input complexity where applicable.
	Software Runtime Environment Security	SRQ3.5.1	Select (Objective)	Software shall run on a host system that is kept up to date with security patches, following industry best practices.
	Standardized Content Display	SRQ3.6.6	Select (Threshold)	The content of displays within a system shall be presented in a consistent, standardized manner.
	Abbreviations	SRQ3.6.5	Select (Threshold)	When abbreviations, mnemonics, or codes are used to shorten data entry, they shall be distinctive and have an intuitive relationship or association to normal language or specific job-related terminology.
	KPP1	SRQ4.1	Select (Threshold)	The resilience module shall provide microgrid resilience analysis for simulations executed within the MPT tool.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
	Operator feedback	SRQ3.6.2	Select (Threshold)	The UI shall inform the operator that input requests are being processed following industry best practices.
	Defined Interface	SRQ1.3.3	Select (Threshold)	The resilience module shall have a defined interface control document to support integration with a digital model.
	Accessibility	SRQ3.6.1	Select (Objective)	Software UI shall conform to Section 508 standard accessibility requirements where applicable.
	Software Reliability	SRQ3.5.2	Select (Objective)	Software backups shall be taken to prevent accidental data loss using a continuous delta backup solution or equivalent.
	Module Description	SRQ3.1.4	Select (Threshold)	Software modules shall provide function description including dependent variables.
	Legacy	SRQ2.1.3	Select (Threshold)	Software shall support legacy software and protocols where possible.
	Nesting	SRQ3.1.3	Select (Threshold)	New code shall be limited to three levels of nesting or less.
	Modification History	SRQ3.1.2	Select (Threshold)	The system shall provide detailed release notes summarizing code changes.
	Name Conventions	SRQ3.1.1	Select (Threshold)	The system shall use appropriate, consistent naming conventions.
	Interface UI	SRQ1.4.1	Defer	Developed application interfaces shall provide an UI to aid data transfer.
	Data Distribution	SRQ1.4	Defer	An application, process, or method shall be created to share data between the digital model and MPT functions.
	Digital Model Integration	SRQ1.2.2	Defer	The digital model shall have a defined interface control document to support integration with the MPT.
	Software Development Best Practices	SRQ3.4.2	Select (Threshold)	Versioning tools shall be used to provide commit history and rationale during development of code.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
	Software Authoring Best Practices	SRQ3.4.1	Select (Threshold)	Software writing shall follow best practices for structure and documentation of code.
	System Resource Reliability	SRQ3.2.1	Select (Threshold)	The Interactive UI shall provide model data and simulation results when requested by the user 99% of the time when the system is normally available.
	Interactive UI Uptime	SRQ3.3.1	Select (Threshold)	The Interactive UI shall be available for use and functional 95% of the time (threshold) and 99% of the time (objective), averaged over 30 days.
	Reuse	SRQ2.1.2	Select (Threshold)	Software shall provide modular and modifiable functions to support ease of integration and code reuse for future development efforts.
	Readability	SRQ3.1.6	Select (Threshold)	Code shall adhere to consistent indentation style, minimize variable length, limit line length, and minimize module length.
	Published APIs	SRQ2.2	Select (Threshold)	The system shall provide a documented list of APIs and how to interact with the system.
	Efficient Code	SRQ3.1.5	Select (Threshold)	Code shall utilize limited computer resourcing.
	Free and Open Standards	SRQ2.1	Select (Threshold)	Software functionality shall use free and open-source tools where possible.
	Data Transfer Mechanism	SRQ1.4.2	Defer	The system shall provide a standardized method for transferring data from the digital model to the MPT resilience module.
	Code Grouping	SRQ3.1.7	Select (Threshold)	Discrete tasks shall be contained in separate code blocks.
<b>SNI.2 - Digital Model</b>				
SNI.2.1 - Create state machine for digital model control module.	Control Module	SRQ1.2.6	Defer	The digital model shall utilize a state machine for the control module.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
SN1.2.2 - Develop digital model system user manual for future efforts.	System User Manual	SRQ3.4.3	Defer	A system user manual shall be developed to document digital model function and capability.
SN1.2.3 - Update battery charge and discharge model.	Battery Library	SRQ1.2.5.3	Defer	The digital model battery discharge / charge cycle library shall be accurate and continuously maintained.
SN1.2.4 - Create dynamic critical and non-critical load function into digital model.	Custom Power Profiles	SRQ1.5.2	Defer	The user shall be able to select critical and non-critical load profiles.
	Load Shedding Profiles	SRQ1.5.3	Defer	The user be able to select a predefined load shedding profile.
SN1.2.5 - Create location-based aero function into digital model.	Wind Data	SRQ1.6.2	Defer	The resilience module shall integrate existing location-based wind data.
	External Geo Location Environment Services	SRQ1.6	Defer	The resilience module shall integrate existing location data.
SN1.2.6 - Create location-based irradiance function into digital model.	Solar Irradiance Data	SRQ1.6.1	Defer	The resilience module shall integrate existing location-based irradiance data.
	External Geo Location Environment Services	SRQ1.6	Defer	The resilience module shall integrate existing location data.
SN1.2.7 - Create location-based climate function into digital model.	External Geo Location Environment Services	SRQ1.6	Defer	The resilience module shall integrate existing location data.
	Climate Data	SRQ1.6.3	Defer	The resilience module shall integrate existing location-based climate data.
SN1.2.8 - Create location-based power consumption function into digital model.	External Geo Location Environment Services	SRQ1.6	Defer	The resilience module shall integrate existing location data.
	Consumption Data	SRQ1.6.4	Defer	The resilience module shall integrate existing location-based, climate dependent energy consumption data.
SN1.2.9 - Calculate resilience metrics within digital model.	KPP2	SRQ4.2	Defer	The resilience module shall perform microgrid resilience analysis for well-defined MSOSA microgrid architectures.



<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
	Recoverability	SRQ1.1.2	Select (Threshold)	The MPT module shall provide recoverability analysis metrics for microgrids.
	Improve DOD Energy Security	SRQ1.1	Select (Threshold)	The Microgrid Planning Tool (MPT) resilience module optimizations shall show an aggregate improvement over traditional methodologies.
	Resilience Module	SRQ1.2.4	Defer	The digital model shall be capable of performing a user selected resilience analysis on the selected model.
	Invulnerability	SRQ1.1.3	Select (Threshold)	The MPT module shall provide invulnerability analysis metrics for microgrids.
	Decision Aid	SRQ1.1.1	Select (Threshold)	The MPT module shall provide resiliency metrics to aid decision making.
SN1.2.10 - Update component and DER models.	DER Library	SRQ1.2.5.2	Defer	The digital model DER library shall be accurate and continuously maintained.
	Component Library	SRQ1.2.5.1	Defer	The digital model component library shall be accurate and continuously maintained.
	Model Source Data	SRQ1.2.5	Defer	The digital model libraries shall be accurate and continuously maintained.
	Battery Library	SRQ1.2.5.3	Defer	The digital model battery discharge / charge cycle library shall be accurate and continuously maintained.
	Control Module	SRQ1.2.6	Defer	The digital model shall utilize a state machine for the control module.
SN1.2.11 - Perform model validation and verification against similar digital models and experimental data.	Digital Model Verification	SRQ1.2.1	Defer	The digital model shall be evaluated against verified experimental data and alternate models.
	Decision Aid	SRQ1.1.1	Select (Threshold)	The MPT module shall provide resiliency metrics to aid decision making.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
SN1.2.12 - Provide disturbance module within digital model.	Disturbance Module	SRQ1.2.3	Defer	The digital model disturbance module shall include multiple use cases and durations.
SN1.3 - Microgrid Planning Tool Module				
SN1.3.1 - Create secure individual user profiles within web tool.	KPP3	SRQ4.3	Defer	The resilience module shall translate well-defined MSOSA microgrid architectures into design parameters parsable by the MPT tool.
	User Profiles	SRQ1.5.4	Defer	The resilience module shall allow user to create secure individual profiles.
SN1.3.2 - Create custom power profiles within web tool.	Predefined Power Profiles	SRQ1.5.1	Select (Threshold)	The user shall be able to select predefined system power profiles.
SN1.3.3 - Create predefined power profiles within web tool.	Load Shedding Profiles	SRQ1.5.3	Defer	The user be able to select a predefined load shedding profile.
	Predefined Power Profiles	SRQ1.5.1	Select (Threshold)	The user shall be able to select predefined system power profiles.
SN1.3.4 - Create custom microgrid components within web tool.	Custom Power Profiles	SRQ1.5.2	Defer	The user shall be able to select critical and non-critical load profiles.
	Load Shedding Profiles	SRQ1.5.3	Defer	The user be able to select a predefined load shedding profile.
SN1.3.5 - Integrate SQL data input within web tool.	Database	SRQ1.5.5	Defer	The resilience module shall utilize databases for persistent data storage.
SN1.3.6 - Simulate system disturbances within web tool.	Disturbance Events	SRQ1.3.1	Select (Objective)	The resilience module shall incorporate global and local disturbance events.
SN1.3.7 - Calculate resilience metrics within web tool.	Resilience Analysis	SRQ1.3.2	Select (Threshold)	The resilience module shall aid resilience analysis.
SN1.3.8 - Provide load shedding capability.	Load Shedding Profiles	SRQ1.5.3	Defer	The user be able to select a predefined load shedding profile.

<b><u>STAKEHOLDER NEED</u></b>	<b><u>SYSTEM REQUIREMENT</u></b>			
	<b>NAME</b>	<b>NUMBER</b>	<b>STATUS</b>	<b>DESCRIPTION</b>
SN1.3.9 - Update component and DER library within web tool.	Source Data	SRQ1.3.4	Defer	The libraries access by the resilience module shall be accurate and continuously maintained.

## APPENDIX C. EXPANDED MPT API DOCUMENTATION

### Microgrid Planning API (0.1.0)

License: [CC0 1.0](#)

## Description and General Usage

### Description

This API provides power load profiles, resilience metrics and corresponding simulated microgrid power generation time series data. A front end client is coming soon. For more information, visit our [Microgrid Planning](#) site. The API conforms to [JSON:API Specification v1.0](#). The following JSON:API media type is used for exchanging data: `application/vnd.api+json`. Include that media type in the `Accept:` header and, when sending data, within the `Content-Type:` header.

## Power Load Profile

### Scenario List

Returns a list of power load scenario profiles available to use.

### Responses

✓ 200 OK

RESPONSE SCHEMA: application/vnd.api+json

---

data      Array of strings

---

GET /load/json

### Response samples

200

**Content type**  
application/vnd.api+json

Copy   Expand all   Collapse all

```
{
  - "data": [
    "air_terminal_3_anonymized"
  ]
}
```

---

## Load Profile

Returns a specific power load profile specified by {scenario\_load}.

### PATH PARAMETERS

---

scenario\_load      string  
*required*      **Default:** "air\_terminal\_3\_anonymized"  
                  **Example:** air\_terminal\_3\_anonymized  
                  Scenario file of grid loads

---

### QUERY PARAMETERS

---

startdatetime	string Default: "2020-09-01 00:00:00" datetime for start of simulation
sim_duration_days	number Default: -1 Example: sim_duration_days=28 Desired duration of simulation, in simulated days. -1 is use native data duration
sim_timeframe_extension_ratio	number $\geq 1$ Default: 1 Example: sim_timeframe_extension_ratio=1 Ratio by which to stretched simulation duration, by repeating timesteps

## Responses

> 200 OK

GET /load/{scenario\_load}/json

### Response samples

200

#### Content type

application/vnd.api+json

#### Example

power-load-collection

Copy Expand all Collapse all

```
{
  - "data": {
    + "output": [ ... ],
    "parameters": { }
  }
}
```

---

# Simulation

## Simulate Power Generation

Returns the simulated power generated for a given set of inputs.

### QUERY PARAMETERS

---

startdatetime	string Default: "2020-09-01 00:00:00" datetime for start of simulation
name <i>required</i>	string Example: name=ABERDEEN REGIONAL ARPT Location of microgrid
dg_power	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Example: dg_power=20 Diesel generator power rating in kilowatts
dg_load	number <input type="text" value="[ 0.1 .. 1 ]"/> Default: <input type="text" value="1"/> Diesel generator load operating ratio.
dg_min_load	number <input type="text" value="[ 0.1 .. 1 ]"/> Default: <input type="text" value="0.6"/> Diesel generator minimum load operating ratio
dg_peak_cons_rate	number <input type="text" value="&gt;= 0"/> Default: "0.1 * power rating" Example: dg_peak_cons_rate=2 Diesel generator peak consumption rate in gallons per hour.
dg_startup_delay	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/>

	Diesel generator startup delay in hours
pv_power	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Example: <input type="text" value="pv_power=288"/> Solar photovoltaic panel power rating in kilowatts
wt_power	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Wind turbine power rating in kilowatts
wt_capacity	number <input type="text" value="[ 0 .. 1 ]"/> Default: <input type="text" value="0.22"/> Ratio of assumed wind to max wind
wt_availability	number <input type="text" value="[ 0 .. 1 ]"/> Default: <input type="text" value="0.98"/> 1 - wind turbine downtime proportion for maintenance
b_discharge_power	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Example: <input type="text" value="b_discharge_power=371.5"/> Power rating for battery energy storage system discharge in kilowatts
b_charge_power	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Example: <input type="text" value="b_charge_power=371.5"/> Power rating for battery energy storage system charge in kilowatts
b_energy	number <input type="text" value="&gt;= 0"/> Default: <input type="text" value="0"/> Example: <input type="text" value="b_energy=743"/> Energy rating for battery energy storage system in kilowatt-hours
b_min_soc	number <input type="text" value="[ 0 .. 1 ]"/> Default: <input type="text" value="0.2"/> Minimum allowable state of charge ratio of battery energy storage system
b_max_soc	number <input type="text" value="[ 0 .. 1 ]"/> Default: <input type="text" value="1"/> Maximum allowable state of charge ratio of battery energy storage system
b_charge_eff	number <input type="text" value="[ 0 .. 1 ]"/> Default: <input type="text" value="0.95"/> 1 - proportion of power lost when charging battery energy storage system
b_discharge_eff	number <input type="text" value="[ 0 .. 1 ]"/>



`b_charge_level` Default: `0.95`  
1 - proportion of power lost when discharging battery energy storage system

---

number `>= 0`  
Default: `"b_energy value"`  
Example: `b_charge_level=743`  
Initial charge level of battery energy storage system in kilowatts

---

#### HEADER PARAMETERS

---

Accept  
*required* string  
Value: `"application/vnd.api+json"`

---

## Responses

> 200 OK

GET `/simulate/json`

### Response samples

200

**Content type**  
application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "data": {
    + "output": [ ... ],
    "parameters": { }
  }
}
```

# Rightsize

## Rightsized Power Generation

Returns rightsized microgrid designs for a fixed set of inputs (paramters are not currently used).

### QUERY PARAMETERS

---

startdatetime	string Default: "2020-09-01 00:00:00" datetime for start of simulation
name <i>required</i>	string Example: name=ABERDEEN REGIONAL ARPT Location of microgrid
dg_load	number [0.1..1] Default: 1 Diesel generator load operating ratio.
dg_min_load	number [0.1..1] Default: 0.6 Diesel generator minimum load operating ratio
dg_startup_delay	number >= 0 Default: 0 Diesel generator startup delay in hours
b_min_soc	number [0..1] Default: 0.2 Minimum allowable state of charge ratio of battery energy storage system
b_max_soc	number [0..1] Default: 1 Maximum allowable state of charge ratio of battery energy storage system
b_charge_eff	number [0..1] Default: 0.95 1 - proportion of power lost when charging battery energy storage system
b_discharge_eff	number [0..1]

Default:

1 - proportion of power lost when discharging battery energy storage system

---

#### HEADER PARAMETERS

---

Accept  
required

string

Value:

---

## Responses

> 200 OK

GET /rightsize/json

### Response samples

200

**Content type**

application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "data": {
    + "output": [ ... ],
    "parameters": { }
  }
}
```

---

## Locations

## Available locations

Returns a dictionary of dictionaries of arrays of available locations by country and state.

### Responses

> 200 OK

GET /locations/json

### Response samples

200

#### Content type

application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "USA": {
    + "SD": [ ... ]
  }
}
```

---

## Name, State, Country

Returns a the name, state and country of a location specified by an {id}.

### PATH PARAMETERS

---

id	integer
required	

Example:

---

## Responses

> 200 OK

GET /locations/{id}/json

### Response samples

200

**Content type**  
application/vnd.api+json

Copy

```
{
  "country": "USA",
  "name": "ABERDEEN REGIONAL ARPT",
  "state": "SD"
}
```

---

## Disturbances

### Available Disturbances

Returns a list of available disturbances available for simulation

## Responses

> 200 OK

GET /disturbances/json

### Response samples

200

**Content type**

application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "data": {
    + "Disturbance Name": [ ... ]
  }
}
```

---

## Grid Makeup

### Query a set of grid energy sources

Obtain stored grid scenarios and energy source makeup of each grid

## Responses

> 200 OK

GET /gridmakeup/json

### Response samples

200

**Content type**

application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "data": {
    + "Scenario Name": [ ... ]
  }
}
```

---

## Resilience

### Generate Resilience Calculations

Run a grid simulation with enhanced parameters and return resilience calculations, statistics, and simulation data

QUERY PARAMETERS

---

startdatetime	string Default: "2020-09-01 00:00:00" datetime for start of simulation
sim_duration_days	number Default: -1 Example: sim_duration_days=28 Desired duration of simulation, in simulated days. -1 is use native data duration
sim_timeframe_extension_ratio	number $\geq 1$ Default: 1 Example: sim_timeframe_extension_ratio=1 Ratio by which to stretched simulation duration, by repeating timesteps
disturbance_datetime	string Default: "2020-09-02 00:00:00" datetime for start of disturbance
name <i>required</i>	string Example: name=ABERDEEN REGIONAL ARPT Location of microgrid
dg_power	number $\geq 0$ Default: 0 Example: dg_power=20 Diesel generator power rating in kilowatts
dg_load	number [0.1 .. 1] Default: 1 Diesel generator load operating ratio.
dg_min_load	number [0.1 .. 1] Default: 0.6 Diesel generator minimum load operating ratio
dg_peak_cons_rate	number $\geq 0$ Default: "0.1 * power rating" Example: dg_peak_cons_rate=2 Diesel generator peak consumption rate in gallons per hour.
dg_startup_delay	number $\geq 0$ Default: 0 Diesel generator startup delay in hours
pv_power	number $\geq 0$



	Default: 0 Example: pv_power=288 Solar photovoltaic panel power rating in kilowatts
wt_power	number $\geq 0$ Default: 0 Wind turbine power rating in kilowatts
wt_capacity	number [0..1] Default: 0.22 Ratio of assumed wind to max wind
wt_availability	number [0..1] Default: 0.98 1 - wind turbine downtime proportion for maintenance
b_discharge_power	number $\geq 0$ Default: 0 Example: b_discharge_power=371.5 Power rating for battery energy storage system discharge in kilowatts
b_charge_power	number $\geq 0$ Default: 0 Example: b_charge_power=371.5 Power rating for battery energy storage system charge in kilowatts
b_energy	number $\geq 0$ Default: 0 Example: b_energy=743 Energy rating for battery energy storage system in kilowatt-hours
b_min_soc	number [0..1] Default: 0.2 Minimum allowable state of charge ratio of battery energy storage system
b_max_soc	number [0..1] Default: 1 Maximum allowable state of charge ratio of battery energy storage system
b_charge_eff	number [0..1] Default: 0.95 1 - proportion of power lost when charging battery energy storage system

b_discharge_eff	number [0..1] Default: 0.95 1 - proportion of power lost when discharging battery energy storage system
b_charge_level	number >= 0 Default: "b_energy value" Example: b_charge_level=743 Initial charge level of battery energy storage system in kilowatts
sim_data_output	boolean Default: true Example: sim_data_output=true Allows omitting return of large output dataset
num_runs	integer >= 1 Default: 1 Example: num_runs=1 number of runs to simulate
scenario_components	string Example: scenario_components=air_terminal_3 scenario name of grid DERs to load from existing json files
disturbance	string Default: "municipal_outage" Example: disturbance=hurricane scenario disturbance to load
scenario_load	string Default: "air_terminal_3_anonymized" Example: scenario_load=air_terminal_3_anonymized Scenario file of grid loads
dg_qty	integer >= 1 Default: 1 Example: dg_qty=1 number of identical diesel generators to create
pv_qty	integer >= 0 Default: 0 Example: pv_qty=1 number of identical photovoltaic generators to create
wt_qty	integer >= 0 Default: 0 Example: wt_qty=1

b\_qty

number of identical wind turbine generators to create

integer `>= 0`

Default: `0`

Example: `b_qty=1`

number of identical battery energy storage devices to create

#### HEADER PARAMETERS

Accept  
**required**

string

Value: `"application/vnd.api+json"`

## Responses

> 200 OK

GET /resilience/json

### Response samples

200

#### Content type

application/vnd.api+json

Copy Expand all Collapse all

```
{
  - "data": {
    + "aggregate_resilience_metrics": { ... },
    + "output_data": [ ... ],
    "parameters": { },
    + "resilience_data": [ ... ],
    "simruntime": "2022/09/05 10:04:13"
  }
}
```

---

# Solar Radiation

## Magnitude of solar radiation

Returns solar radiation array where timestamp and magnitude are provided in an inner array, and timestamp is specified by an array [hours, minutes, seconds, milliseconds], given an input location {id} and {date} (MM-DD format).

### PATH PARAMETERS

---

id required	integer Example: 726590
date required	string Example: 09-12

---

## Responses

> **201** Created

GET /solar/{id}/{date}/json

### Response samples

**201**

**Content type**  
application/vnd.api+json

```
[  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    0  
  ],  
  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    0  
  ],  
  - [  
    + [ ... ],  
    229.5  
  ],  
  - [  
    + [ ... ],  
    324.33  
  ],  
  - [  
    + [ ... ],  
    536.5  
  ],  
]
```

- [ + [ ... ],  
662.67  
],  
- [ + [ ... ],  
704.17  
],  
- [ + [ ... ],  
735.17  
],  
- [ + [ ... ],  
638  
],  
- [ + [ ... ],  
537.33  
],

- [ + [ ... ],  
456.33  
],  
- [ + [ ... ],  
298.5  
],  
- [ + [ ... ],  
134.33  
],  
- [ + [ ... ],  
7.67  
],  
- [ + [ ... ],  
0  
],

- [  
+ [ ... ],  
0  
,  
- [  
+ [ ... ],  
0  
,  
- [  
+ [ ... ],  
0  
,  
- [  
+ [ ... ],  
0  
]  
]

# APPENDIX D. EXAMPLE SCRUM DETAILS

## Sprint 6 (7/19 – 8/2)

Thursday, April 28, 2022 9:22 AM

### Planning

#### Goal

*What is the primary goal of the sprint? What will be complete at the end?*

🏠 Return a resilience calculation based on a disturbance. Improve analysis narrative in report draft.

#### User Stories

*What customer facing tasks will we complete?*

- ★ Implement disturbance modeling in python code
  - What even are probabilities? (How are probabilities calculated from json file?)
  - Why do disturbances not appear random? (same 1st run each time)
- ★ Implement more than one of a type of DER in python code
  - Implement 2 independent diesel generators - Tom to describe this as a disturbance range.
- ★ Implement baseline resilience calculations
  - Determine disturbance start and end times (or allow for user to provide start and end time)
  - Determine pre-disturbance power delivered (for invulnerability)
  - Calculate invulnerability
  - Calculate recoverability
  - Calculate resilience
- ★ Update Chapter 3 of report
  - Beginning of analysis discussion

#### Internal/Individual tasks

*What internal team tasks need to be completed? These are important for the team to progress, but may not be directly customer facing.*

- Look at 5200.46 - MS-VVA
- Clean up Chapter 1-2 of report
  - Still needs re-org

### Retrospective

#### What went well?

- Worked through 3-4 members of team on TDY
- Completed sprint goal for the most part
- Sprint goal aligned with schedule

#### What needs improvement?

- Took until halfway through sprint to populate task board
- Team cohesiveness was low, did not make full effort to find times to meet as a group
- Did not work on report as much as desired

#### Actions

- Update Ch2 and 3 in report with updated SE diagrams
- Cleanup actions assigned to Dave on calculation methods

Figure 60. Example Sprint Planning and Retrospective Board



THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF REFERENCES

- Agile Alliance. n.d. "What Is Agile? Agile 101." Accessed April 16, 2022. <https://www.agilealliance.org/agile101/>.
- Anderson, William W. 2020. "Resilience Assessment of Islanded Renewable Energy Microgrids." PhD diss, Naval Postgraduate School. <http://hdl.handle.net/10945/66574>.
- Anglani, Norma, Giovanna Oriti, Ruth Fish, and Douglas L. Van Bossuyt. 2021. "Design and Optimization Strategy to Size Resilient Stand-Alone Hybrid Microgrids in Various Climatic Conditions." In *2021 IEEE Energy Conversion Congress and Exposition (ECCE)*, 210–17. Vancouver, BC, Canada: IEEE. <https://doi.org/10.1109/ECCE47101.2021.9595049>.
- Anuat, Edward, Douglas L. Van Bossuyt, and Anthony Pollman. 2021. "Energy Resilience Impact of Supply Chain Network Disruption to Military Microgrids." *Infrastructures* 7 (1): 4. <https://doi.org/10.3390/infrastructures7010004>.
- Atlassian. n.d. "What Is Agile?" Accessed October 13, 2022. <https://www.atlassian.com/agile>.
- Atlassian Bitbucket. n.d. "Git Feature Branch Workflow." Accessed October 13, 2022. <https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow>.
- Bickford, Jason, Douglas L. Van Bossuyt, Paul Beery, and Anthony Pollman. 2020. "Operationalizing Digital Twins Through Model-Based Systems Engineering Methods." *Systems Engineering* 23 (6): 724–50. <https://doi.org/10.1002/sys.21559>.
- Blanchard, Benjamin S., and W. J. Fabrycky. 2011. *Systems Engineering and Analysis*. 5th ed. Prentice Hall International Series in Industrial and Systems Engineering. Boston: Prentice Hall.
- Bolen, Curtis D, Victoria Chu, Andy Q Dang, Paul T Kim, Christian Proctor, and Bridget R Shideler. 2021. "Integrating Power-Flow, Resilience, and Cost Models for Naval Installation Microgrids." Systems engineering capstone report, Naval Postgraduate School. <http://hdl.handle.net/10945/67110>.
- Buede, Dennis M. 2009. *The Engineering Design of Systems*. 2nd ed. Hoboken, NJ: John Wiley and Sons, Inc.

- Chief Information Officer, U.S. Department of Defense. n.d. “DoDAF Viewpoints and Models. Operational Viewpoint. OV-1: High Level Operational Concept Graphic.” Accessed October 13, 2022. [https://dodcio.defense.gov/Library/DOD-Architecture-Framework/dodaf20\\_ov1/](https://dodcio.defense.gov/Library/DOD-Architecture-Framework/dodaf20_ov1/).
- Claire, Nicolas, Jun Rentschler, Albertine Potter van Loon, Sam Oguah, Amy Schweikert, Mark Deinert, Elco Koks et al. 2019. *Stronger Power: Improving Power Sector Resilience to Natural Hazards*. Washington, DC: The World Bank. [https://www.researchgate.net/publication/341553182\\_STRONGER\\_POWER\\_Improving\\_Power\\_Sector\\_Resilience\\_to\\_Natural\\_Hazards](https://www.researchgate.net/publication/341553182_STRONGER_POWER_Improving_Power_Sector_Resilience_to_Natural_Hazards).
- Cornell, Phillip. 2009. “Energy and the Three Levels of National Security: Differentiating Energy Concerns within a National Security Context.” *Connections* 8 (4): 63–80. <http://www.jstor.org/stable/26326186>.
- Cullom, RADM Philip H. 2009. “Energy Security: Strategic Overview.” Slides, October 14, 2009. <https://www.nre.navy.mil/media/document/radm-cullom-energy-security-overviewpdf>.
- Department of Defense. 2009. *Installation Energy Management*. DOD Instruction 4170.11. Washington, DC: Department of Defense. <https://www.acq.osd.mil/dodsc/library/dodi-4170-11-installation.pdf>.
- Francis, Royce, and Behailu Bekera. 2014. “A Metric and Frameworks for Resilience Analysis of Engineered and Infrastructure Systems.” *Reliability Engineering and System Safety* 121: 90–103. <https://doi.org/10.1016/j.ress.2013.07.004>.
- Giachetti, Ronald E., Douglas L. Van Bossuyt, William Anderson, and Giovanna Oriti. 2022. “Resilience and Cost Trade Space for Microgrids on Islands.” *IEEE Systems Journal* 16 (3): 3939–49. <https://doi.org/10.1109/JSYST.2021.3103831>.
- Giachetti, Ronald E., Christopher J. Peterson, Douglas L. Van Bossuyt, and Gary W. Parker. 2020. “Systems Engineering Issues in Microgrids for Military Installations.” *INCOSE International Symposium* 30 (1): 731–46. <https://doi.org/10.1002/j.2334-5837.2020.00751.x>.
- GitLab. n.d. “Squash and Merge.” Accessed October 25, 2022. [https://docs.gitlab.com/ee/user/project/merge\\_requests/squash\\_and\\_merge.html](https://docs.gitlab.com/ee/user/project/merge_requests/squash_and_merge.html).
- Grussing, Michael N., Samuel L. Hunter, Steve Gunderson, Mary Canfield, Ed Falconer, and Albert Antelman. 2010. *Development of the Army Facility Mission Dependency Index for Infrastructure Asset Management*. ERDC/CERL TR-10-18. Champaign, IL: Construction Engineering Research Laboratory. <https://apps.dtic.mil/sti/pdfs/ADA552791.pdf>.

- Hildebrand, Joshua P. 2020. "Estimating the Life Cycle Cost of Microgrid Resilience." Master's thesis, Naval Postgraduate School. <http://hdl.handle.net/10945/66658>.
- Hirsch, Adam, Yael Parag, and Josep Guerrero. 2018. "Microgrids: A Review of Technologies, Key Drivers, and Outstanding Issues." *Renewable and Sustainable Energy Reviews* 90 (March): 402–11. <https://doi.org/10.1016/j.rser.2018.03.040>.
- Innoslate. n.d. "Innoslate – Systems Engineering and Requirements Management Software." Accessed October 21, 2022. <https://www.innoslate.com/>.
- Kain, Alissa, Douglas L. Van Bossuyt, and Anthony Pollman. 2021. "Investigation of Nanogrids for Improved Navy Installation Energy Resilience." *Applied Sciences* 11 (9): 1–30. <https://doi.org/10.3390/app11094298>.
- Kujawski, Edouard, and Gregory Miller. 2009. "The Mission Dependency Index: Fallacies and Misuses." In *INCOSE International Symposium*, 19:1565–80. <https://doi.org/10.1002/j.2334-5837.2009.tb01035.x>.
- Kukhnavets, Pavel. n.d. "Sprints." Accessed October 13, 2022. <https://hygger.io/guides/agile/scrum/sprints-in-scrum/>.
- Naval Facilities Engineering Systems Command. 2021. "3 Pillars of Energy Security (Reliability, Resilience, & Efficiency)." Washington Navy Yard, DC: Naval Facilities Engineering Systems Command. <https://www.wbdg.org/ffc/navy-navfac/p-publications/p-602>.
- Office of the Assistant Secretary of Defense for Energy, Installations and Environment. 2015. *2016 DOD Operational Energy Strategy*. Washington, DC: Office of the Assistant Secretary of Defense for Energy, Installations and Environment. <https://www.acq.osd.mil/eie/Downloads/OE/2016%20DoD%20Operational%20Energy%20Strategy%20WEBc.pdf>.
- Patel, Sandip, and Jigish Zaveri. 2010. "A Risk-Assessment Model for Cyber Attacks on Information Systems." *Journal of Computers* 5 (3): 352–59.
- Pedersen, Kjartan, Jan Emblemvag, Reid Bailey, Janet Allen, and Farrokh Mistree. 2000. "Validating Design Methods & Research: The Validation Square." In *2000 ASME Design Engineering Technical Conferences*, 1–13. Baltimore, MD: ASME Design Engineering Technical Conferences. [https://www.researchgate.net/publication/238355807\\_The%27%27Validation\\_Square%27%27-Validating\\_Design\\_Methods](https://www.researchgate.net/publication/238355807_The%27%27Validation_Square%27%27-Validating_Design_Methods).
- Peterson, Christopher J., Douglas L. Van Bossuyt, Ronald E. Giachetti, and Giovanna Oriti. 2021. "Analyzing Mission Impact of Military Installations Microgrid for Resilience." *Systems* 9 (3): 1–19. <https://doi.org/10.3390/systems9030069>.

- Reich, Daniel. n.d. "A Suite of Tools." Migrogrid Planning Tools. Accessed October 8, 2022. <https://microgrid.nsetti.nps.edu/>.
- Reich, Daniel, and Giovanna Oriti. 2021. "Rightsizing the Design of a Hybrid Microgrid." *Energies* 14 (14): 4273. <https://doi.org/10.3390/en14144273>.
- Siritoglou, Petros, Giovanna Oriti, and Douglas L. Van Bossuyt. 2021. "Distributed Energy-Resource Design Method to Improve Energy Security in Critical Facilities." *Energies* 14 (May): 2773. <https://doi.org/10.3390/en14102773>.
- SouthWest Water Company. 2021. "Preparing for a Water Emergency During a Hurricane." *SouthWest Water Company (Blog)* (blog). June 24, 2021. <https://swwc.com/preparing-for-a-water-emergency-during-a-hurricane/>.
- TONEX. n.d. "Microgrid Training." Accessed October 14, 2022. <https://www.tonex.com/training-courses/microgrid-training/>.
- United States Navy and Marine Corps. 2020. *United States Navy and Marine Corps Digital Systems Engineering Transformation Strategy*. Washington, DC: United States Navy and Marine Corps. <https://nps.edu/documents/112507827/0/2020+Dist+A+DON+Digital+Sys+Eng+Transformation+Strategy+2+Jun+2020.pdf>.

## INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center  
Ft. Belvoir, Virginia
2. Dudley Knox Library  
Naval Postgraduate School  
Monterey, California



## DUDLEY KNOX LIBRARY

NAVAL POSTGRADUATE SCHOOL

[WWW.NPS.EDU](http://WWW.NPS.EDU)

---

WHERE SCIENCE MEETS THE ART OF WARFARE