An Autonomous Energy Management Platform for Resilient Operation of MicroGrids

Kiyoshi Nakayama and Ratnesh Sharma
Dept. of Energy Management, NEC Laboratories America, Inc.
Cupertino, CA 95014, USA
Email: {knakayama, ratnesh}@nec-labs.com

Abstract—**In this paper, we design an architecture of Energy Management Systems (EMS) for resilient operation of microgrids and describe autonomous mechanisms that dynamically enable 1) enhancement for scalability, 2) system modification and upgrade, and 3) failure detection and recovery. Designing the resilient architecture and achieving autonomous mechanisms are of critical importance for a system operator to be able to run the EMS in a flexible and stable manner as any maintenance or system failure should not cause costly damage to the grid vendor. The focus of this paper is on the communications aspect to distribute core functionality of Microgrid EMS into different components with several servers so that system upgrade or any failure of the EMS components would not affect the other modules and be handled without interrupting other grid operations and services. Based on the distributed architecture of EMS, we propose a synchronization mechanism of autonomous functions, which are dynamic micro-grid configuration, failure/outlier detection, and system modification and recovery. We validate those mechanisms using the platforms in NEC Labs and Amazon Web Services (AWS) with experimental results on system modification and failure recovery time.**

I. INTRODUCTION

An operator at a utility or aggregator needs to monitor and manage a microgrid and its devices (e.g. battery, photovoltaics (PV), load, diesel generator) with Energy Management Systems (EMS) that can optimize, control, and scale up the grid. In addition, the microgrid operator will have to develop and deploy multiple microgrids within a region under their authority. Energy storage vendors received nearly 40 percent of grid edge venture investment in 2015, which is more than any other grid edge technology submarket [1]. NEC Labs has been leading research on microgrid EMS focusing on battery optimization and resilient control of a microgrid [2], [3], [4]. The outcomes of the project provide an efficient microgrid management scheme by accomplishing the task to optimize the microgrid's performance by defining long-term directives or control strategies. The management system is also able to operate and control the microgrid in real time and satisfy all operational constraints. The architecture presented in the EMS has advisory and real-time layers within one server. While most of the microgrids support a centralized architecture with all the necessary functions implemented within a server [5], it is inevitable to distribute the microgrid management functions based on the different nature of controls as in advisory layer and real-time layer.

In designing and creating such a distributed EMS platform, we need to address the problems of enhancing, maintaining, upgrading, and recovering the multiple EMS components in real time without affecting or stopping the other systems that are already running, all of which are defined as autonomous resilient operations of multiple microgrids and devices. For instance, when adding a microgrid controller to a currently running EMS platform, its configuration and system information should automatically be registered in the database that is accessed by an operational platform to enhance the microgrid management systems and sustain their controls. Failures that may happen in microgrids or to their systems should be quickly detected and, if possible, restored autonomously based on recovery mechanisms using back-up devices. Proposing dynamic and autonomous schemes for failure recovery is crucial to minimize the cost caused from the interruption of the running microgrids and systems.

In this context, this paper describes an EMS architecture that realizes resilient operation of microgrids and proposes an autonomous EMS operational platform that enables dynamic mechanisms for scalable, flexible, and fault-tolerant operation. By synchronizing EMS components' state information exploiting a centralized scalable database that can also store a large amount of measurement data reported from a number of microgrid devices, the proposed platform dynamically aggregates and manages the resilience controllers that balance the power supplies and demands of microgrid devices during its real-time operation. The mechanisms implemented in the autonomous EMS platform lead to minimizing the operational cost of setting up, enhance, update, and recovering the management systems without stopping and/or interrupting the running systems, which significantly benefits both customers and utility operators.

II. RELATED WORK

The dynamic configuration has mainly been discussed in the domain of networks and distributed systems [6] and applied to cloud-based environment [7], configuration in complicated topological network [8], among others. However, the dynamic configuration and operational mechanism in application to microgrid management has not been discussed much in literature, although centralized architectures targeting single microgrid control [5] have been intensively addressed. In the traditional centralized architectures, a microgrid controller has been installed for managing an entire grid to ensure resilient control of its devices such as distributed energy storage systems (ESS) and Photovoltaics (PV). However, setting up multiple microgrids in an autonomous manner that leads to efficient update, enhance, recovery, and smooth maintenance of the systems should be addressed so that we could minimize the cost of operation without hurting its operations.

Coordination and management of multiple microgrids as an EMS service is relatively an emerging idea to scale up the deployment of microgrids. Multi-agent approaches to manage multiple microgrids have been discussed so far to control complicated systems with minimum data exchange and computation demands [9]. Agent could be a physical or virtual entity for each microgrid or its device(s) with certain level of autonomy where agents communicate with each other. What is needed for operators, however, is the consistent mechanism to synchronize all the configuration information and system status in an easily accessible manner at any time. Information of all the microgrids could be configured using central EMS Operational Platform that provides consistent management of all the local controllers and devices. The similar architecture is discussed in [10], [11], although the multi-agent system discussed in the literature is focused on optimizing and controlling aspects with multiple microgrids, not on the operational and configuration aspects.

Analysis on the impact of communication failures on power grids has been conducted recently [12], [13], [14]. This is the boundary area that tries to extract the impact related to communications failure and intensively discussed these days, which gives rise to a new restoration model that minimizes the impact of failed communications on EMS and microgrids.

Our autonomous framework digs into the mechanisms to set up, enhance, update, and recover microgrids and their energy devices in the EMS operational platform (such as a facilitator platform discussed in literature [10]). In particular, our synchronization mechanisms using central EMS database accessed by Dynamic Operation Handler, Failure/Outlier Detector, and System Recovery Module as well as microgrid-side configuration and operation mechanisms are novel and emerging techniques with the advent of cloud virtualization and IoT (Internet of Things) devices since the system state (e.g. active, not-active, failed, updated) in the database can always be monitored and updated by diagnostic and recovery engines in an orderly fashion and utilized by the dynamic operation engine in the EMS platform.

III. EMS Architecture and System Design

In this section, we describe the architecture and design of our EMS by enabling communications among modules and compare it with a traditional centralized EMS model.

A. Traditional Centralized EMS Architecture

The overview of the previous EMS architecture can be found in Figure 1 where all the functions (e.g. secondary control [13], management modules) are implemented within one EMS server. The main functions are listed as follows:

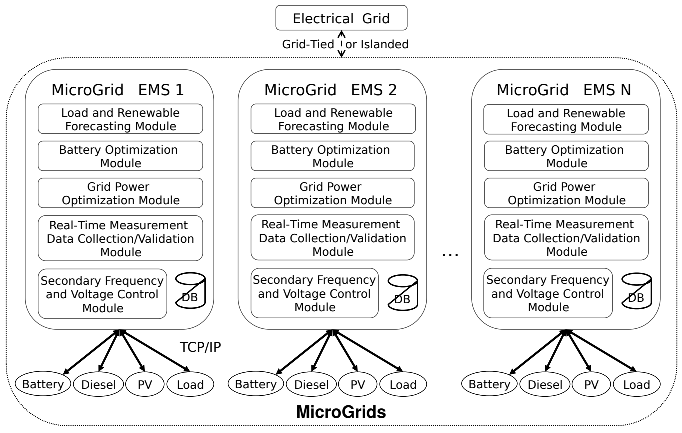

Electrical Grid
Grid-Tied or Islanded
MicroGrid EMS 1
MicroGrid EMS 2
MicroGrid EMS N
Load and Renewable Forecasting Module
Battery Optimization Module
Grid Power Optimization Module
Real-Time Measurement Data Collection/Validation Module
Secondary Frequency and Voltage Control Module
DB
...
TCP/IP
Battery
Diesel
PV
Load
MicroGrids

Fig. 1. Traditional centralized EMS architecture.

- Load and renewable generation forecasting
- Cost optimization for battery charging/discharging and grid power provision
- Real-time measurement data collection from microgrid devices and its validation
- Secondary control for frequency and voltage regulations

The Primary Frequency Control (Droop Control) resides in each microgrid resource such as battery and diesel generator [15] to respond to the changes in power within milliseconds to seconds.

The issues of having this type of architecture mainly lie in resiliency and scalability. For instance, the applications failure in forecasting module could affect the other critical module of frequency and voltage control. As primary and secondary controls need to respond to changes within milliseconds to seconds, it is not ideal to combine all the applications within one physical server to conduct resilient microgrid operations. In addition, in this architecture, one microgrid does not get data from other microgrids as the control, optimization, and management are independent from each other.

Therefore, what is important for the resilient operation of microgrids is to design an architecture where the critical modules (reai-time layer) and other optimization and management modules (advisory layer) are detached and interact with each other. In addition, primary and secondary controls should be installed locally and less affected by applications upgrade by the other optimization and management modules.

B. Resilient Architecture with Distributed EMS Functions

An architecture designed based on communications among modules is found in Figure 2. Here, we have clearly separated the functionality so that EMS consists of the following four components. The architecture employs the facilitator-type of system architecture described in [9], [16] so that optimization can be conducted in an EMS server by using powerful and reliable computation platform such as public clouds. By supporting a facilitator-type or aggregator-type architecture, EMS can aggregate Distributed Energy Resources (DERs) and energy storages of multiple microgrids where optimization and operation based on the correlation among microgrids are feasible.

1) Operational Platform (OP): Operational Platform is introduced in this architecture in order to make the EMS operations more resilient. The autonomous mechanisms for system upgrade, maintenance, enhancement, and failure recovery could be implemented in OP, and the details are described in the section IV.

2) Management Engine (ME): While RC aggregates the microgrid devices that require controls with seconds, Management Engine (ME) aggregates DERs to conduct economic optimization and dispatches management commands to RCs. The detail of its functionality can be found in [2], [3]. It basically receives the input of historical data on load and renewable profiles and conducts forecasting and generate optimum battery profile based on the electricity price and battery degradation.

3) EMS Database (EMSDB): All the configuration information, system status, measurement data, forecast results, optimization commands are stored in the EMS Database that can be accessed by OP and ME. EMS database also keeps track of all the configuration information, system conditions, and diagnostic results. This makes the operation of multiple microgrids less complicated and easier rather than just having state information in each microgrid that can be exchanged by messaging.

4) Resilience Controller (RC): Resilience Controller (RC) mainly realizes the real-time control to adjust the balancing of demand and supply. It receives the real-time measurement data in order to balance the power and restore the nominal frequency. RC aggregates the microgrid devices with primary control and makes the grid stable and reliable at all times. In particular, RC has the Secondary Frequency Control (AGC) function to constantly adjust the setpoints of generators to balance the power every second(s). After a RC receives the the commands dispatched by ME for economic cost optimization, RC verifies whether it can follow the commands or not by looking at balancing factor of supply and demand. If it does not cause adverse affect on frequency regulation, RC dispatches the commands to the microgrid devices, otherwise RC modifies the commands to sustain the grid operation.

C. Necessary Functions to be Implemented

While this new architecture provides us with flexibility in updating and enhancing EMS, we need to address following aspects.

- Quick recovery from failure in Management Engine as well as Resilience Controller(s)
- Communications delay in dispatching operational and optimization commands to Resilience Controllers

From the next section, we address those aspects by introducing operational platform that handles resilient operations.

IV. Autonomous EMS Operational Platform

As the functions of management engines are described in [2], [3], [4], we introduce an autonomous EMS Operational Platform in this section.

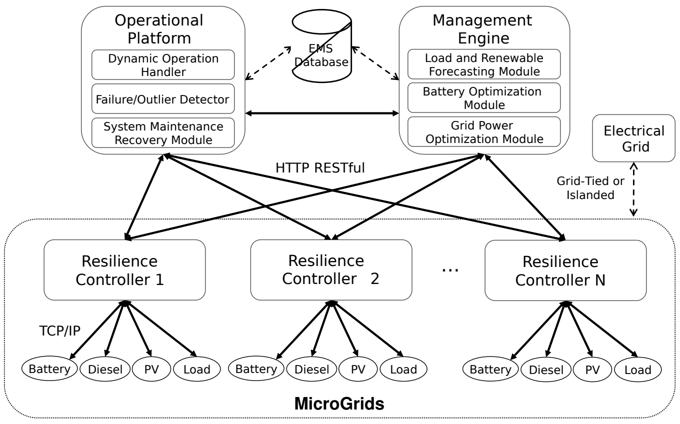


Fig. 2. Resilient EMS Architecture with Operational Platform.

A. Dynamic Operation Handler

Dynamic Operational Handler constantly keeps track of the configuration status of microgrids and checks whether any microgrid is added or removed from the EMS Database. It also keeps track of the communications and system status of microgrids and uses the updated status information in dispatching operational and optimization commands to the RCs.

A1) MicroGrid Configuration Status Tracker: This module consists of 3 key functions, which are Configured MicroGrid Extractor Info Checker, Network Connection Tracker, and System Status Tracker. The unique feature of this module is to constantly keep track of the configuration information, network connection status, and system status of microgrids and devices that have been uploaded by local resilience controller to the EMS Database via Operational Platform so that the results of the tracked data are utilized in the following optimization command dispatcher.

A2) Optimization Command Dispatcher: The command dispatcher mainly consists of Battery Degradation Analyzer and Command Handler. Battery Degradation Analyzer calculates the expected life of battery based on the expected usage pattern such as charging rate behavior. The important mechanism on top of those analyzers and trackers is Command Handler, which copes with requesting and receiving optimization commands from management engine and constantly reflects the updated results of dynamic configuration and system status on the commands constructed in the other modules such as the management engine.

B. Failure/Outlier Detector

Failure/Outlier Detector consists of two functions of extracting failures and/or outliers in management engines or microgrids, which are Data Analytics for System Outlier Detection and Diagnostic Polling.

B1) Data Analytics for System Outlier Detection: The primary way to detect the outlier condition in microgrids is to analyze the measurement data uploaded by resilience controllers to extract abnormal behaviors, which is mainly

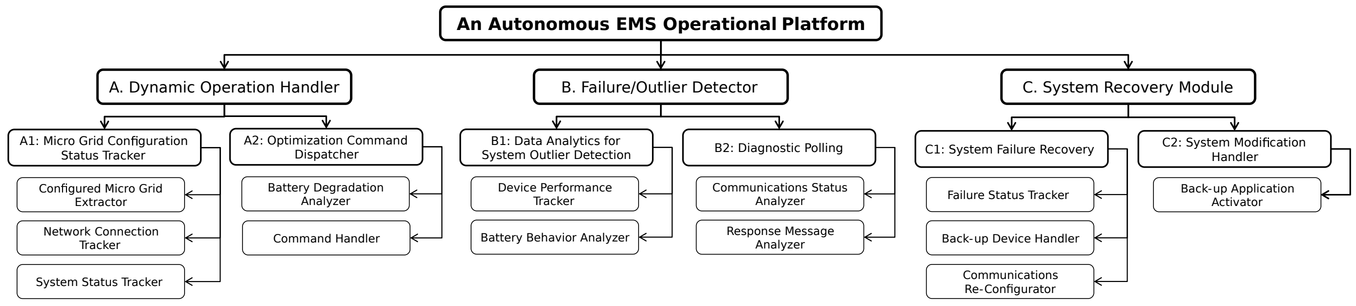

An Autonomous EMS Operational Platform
A. Dynamic Operation Handler
B. Failure/Outlier Detector
C. System Recovery Module
A1: Micro Grid Configuration Status Tracker
A2: Optimization Command Dispatcher
B1: Data Analytics for System Outlier Detection
B2: Diagnostic Polling
C1: System Failure Recovery
C2: System Modification Handler
Configured Micro Grid Extractor
Network Connection Tracker
System Status Tracker
Battery Degradation Analyzer
Command Handler
Device Performance Tracker
Battery Behavior Analyzer
Communications Status Analyzer
Response Message Analyzer
Failure Status Tracker
Back-up Device Handler
Communications Re-Configurator
Back-up Application Activator

Fig. 3. Autonomous Functions of Operational Platform.

conducted by Device Performance Tracker. Battery Behavior Analyzer keeps track of the behavior of batteries such as state of charge in order to guarantee that they are effectively used or there is not any unexpected leaking happening during microgrid operations.

B2) Diagnostic Polling: The diagnostic polling mechanism comprises Communications States Analyzer and Response Message Analyzer. Both of the analytical methods are aligned with the failure detection proposed in the area of network communications and distributed systems (e.g. [17]) although the analytical schemes are classified into specific categories defined by the components of EMS for response message analysis. The format and protocol (e.g. XML schema, response code) should be predefined among EMS Operational Platform, Management Engine, and microgrids so that the Operational Platform can precisely capture the system states. After the detection of failure, the mechanism updates the system status in EMS database and the status info is synchronized and utilized by the other modules in Operational Platform and Management Engine.

C. System Recovery Module

System Recovery Module is synchronized with other functions such as failure detectors that reflect the updated configuration and system status of microgrids and devices on the record of EMS Database. This module realizes automatic failure recovery and modifying systems based on the back-up devices and applications. The mechanism to switch from a main system to a back-up system is a key application to restore systems without any interruptions, which leads to stable and resilient management and operation of microgrids.

C1) System Failure Recovery: On the basis of the outlier analysis and polling results of the EMS whose system status is constantly updated in the EMS database, Operational Platform could quickly be notified of any failure by real-time system condition tracking mechanism called Failure Status Tracker. Back-up Device Handler utilizes back-up devices to restore the failure whose autonomous mechanism is explained in next section. In addition, this module has Communications Reconfiguration function to automatically switch the communications gateway to which the optimization and operational commands are sent. This requires the reconfiguration of HTTP server and client information including IP Address, Server Port, etc. Once all the communications information is set up, the module enables other properly running systems to communicate with back-up device(s) to send operational commands.

C2) System Modification Handler: This is the mechanism to update and/or maintain management engine or resilience controller(s) without hurting the current operations. The key concept on Back-up Application Activator is to check system and communications status of all the back-up applications and send the notification to other modules to notify the system modification event(s). If the EMS's component status is "Updated", the Operational Platform is switching the running application to its duplicate one. This is an essential function to conduct commissioning-phase operation to make sure the updated application is error-free and ready for the use in real operations.

V. Autonomous Mechanisms for Resilient Operations

Key features listed in section IV are dynamic and conducted in parallel. Here, we describe how those modules are synchronized in Operational Platform and handle unexpected issues during daily operations. The summary of the essential features that solve the important problem is as follows:

1) Thread computing for synchronizing configuration and system status is the key point for the dynamic operations in order to update all the configuration and system running statuses, which are all used in planning, scheduling, and dispatching operation and optimization commands. The Operational Platform is able to quickly detect the updated status and conducts appropriate procedure to modify the system or commands by constantly looking into the EMS database to grasp the change in each component's configuration or system status as EMS database is designed to precisely capture them in a real-time manner.

2) Combination of autonomous outlier detection, system recovery mechanism, and dynamic operation based on the updated system status practically enables consistent operation, optimization, and restoration, all of which are critical to management of EMS and beneficial when scaling up EMS having a number of micro grids installed. Cost of constantly monitoring the system status by operators can be reduced because of the autonomous nature of failure restoration and systems modification mechanisms.

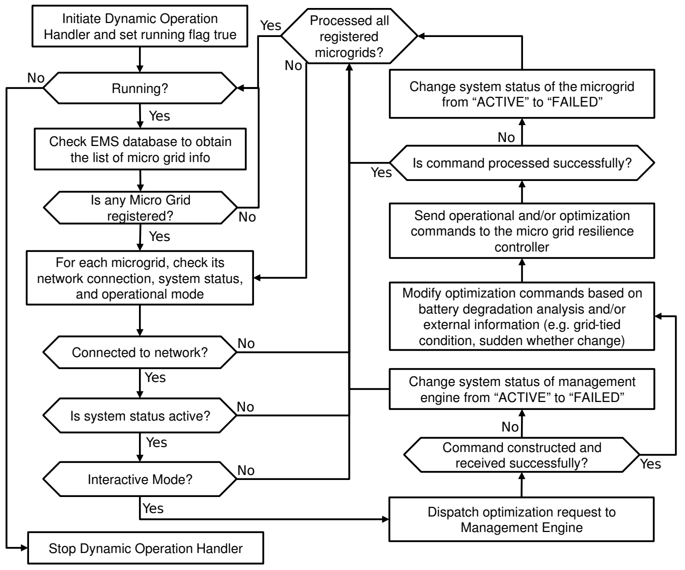

Initiate Dynamic Operation Handler and set running flag true
Running?
No
Yes
Check EMS database to obtain the list of micro grid info
Is any Micro Grid registered?
No
Yes
For each microgrid, check its network connection, system status, and operational mode
Connected to network?
No
Yes
Is system status active?
No
Yes
Interactive Mode?
No
Yes
Stop Dynamic Operation Handler
Processed all registered microgrids?
Yes
No
Change system status of the microgrid from "ACTIVE" to "FAILED"
No
Is command processed successfully?
Yes
Send operational and/or optimization commands to the micro grid resilience controller
Modify optimization commands based on battery degradation analysis and/or external information (e.g. grid-tied condition, sudden whether change)
Change system status of management engine from "ACTIVE" to "FAILED"
No
Command constructed and received successfully?
Yes
Dispatch optimization request to Management Engine

Fig. 4. Procedure of Dynamic Configuration and Operation.

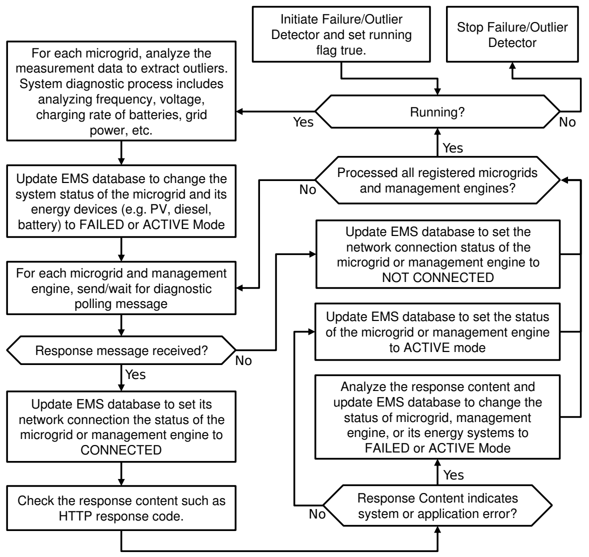

Initiate Failure/Outlier Detector and set running flag true.
Stop Failure/Outlier Detector
For each microgrid, analyze the measurement data to extract outliers. System diagnostic process includes analyzing frequency, voltage, charging rate of batteries, grid power, etc.
Running?
Yes
No
Yes
Processed all registered microgrids and management engines?
No
Update EMS database to change the system status of the microgrid and its energy devices (e.g. PV, diesel, battery) to FAILED or ACTIVE Mode
Update EMS database to set the network connection status of the microgrid or management engine to NOT CONNECTED
For each microgrid and management engine, send/wait for diagnostic polling message
Update EMS database to set the status of the microgrid or management engine to ACTIVE mode
Response message received?
No
Yes
Analyze the response content and update EMS database to change the status of microgrid, management engine, or its energy systems to FAILED or ACTIVE Mode
Update EMS database to set its network connection the status of the microgrid or management engine to CONNECTED
Yes
Response Content indicates system or application error?
No
Check the response content such as HTTP response code.

Fig. 5. Procedure of Failure and Outliers Detection.

A. Dynamic Configuration and Operation Mechanism

The procedure of dynamic configuration and operation is depicted in Figure 4. The EMS needs to be initiated in the following order: EMS Database, Operational Platform, Management Engine, and Resilience Controller(s). EMS configuration information is registered at the time the system is initiated as its automatic registration mechanism sends the info of its configuration to Operational Platform, and then the platform registers it in EMS Database. Once EMS information including microgrid(s) is configured in the EMS Database, Dynamic Operation Engine tries to manage and optimize the microgrid(s) to which commands for grid control are sent. As network connection status, system status, and operational mode of the RCs are constantly updated by the other functions, this module utilizes the updated information to request or dispatch commands.

B. Failure and Outliers Detection Mechanism

This procedure as in Figure 5 describes the autonomous failure and/or outlier detection mechanism so that each microgrid or management engine processes diagnostic polling messages sent from or to Operational Platform to see whether they are properly working or not. If any abnormal behavior that brings adverse affect to EMS would be detected before sending the polling diagnostic messages and reflected on the system status recorded by EMS Database. After a RC of the microgrid initiates the Failure Detection module, it updates the system status of itself and its devices in the EMS Database so that Operational Platform can start back-up systems to recover the failure. There are two types of failures, which are communications failure and application failure. In particular, applications failure can be detected by analyzing the content of response message of diagnostic polling such as the response code of HTTP message. This is the autonomous mechanism to extract the system condition of microgrids and/or management engine, and update their configuration information in EMS database as well as state information. Operational Platform then handles the results by diagnostic polling from/to the microgrids and management engine.

C. System Recovery Mechanism

Once the outlier of failure condition is detected, EMS Operational Platform activates System Recovery mechanism using back-up device or application as shown in Figure 6. In order to make the system recovery mechanism function, we define the two types of following state information as follows:

- Failed: A microgrid is being failed by unexpected incident/outlier
- Updated: A microgrid is being upgraded, maintained, or modified by system operators or engineers

After checking that its backup system is appropriately configured and connected to the network, it starts the recovery process in both Operational Platform and the back-up device(s). The configuration information and its system status are all synchronized with the EMS database accessed by Operational Platform, Management Engine, and Resilience Controllers so that once the information of microgrids or their devices is registered during the system initialization, Operational Platform can detect the change of those modules in EMS database and conduct appropriate procedure to manage and optimize the microgrids.

VI. Validation

We have implemented the modules and mechanisms introduced in sections V to validate the system behaviors.

The Operational Platform (OP), Management Engines (ME), and EMS Storage (Database) are installed in the platforms of Amazon Web Services (AWS) and Resilience Controllers (RCs) are installed in the servers within NEC's Intranet. By using AWS platform, vendors could reduce the

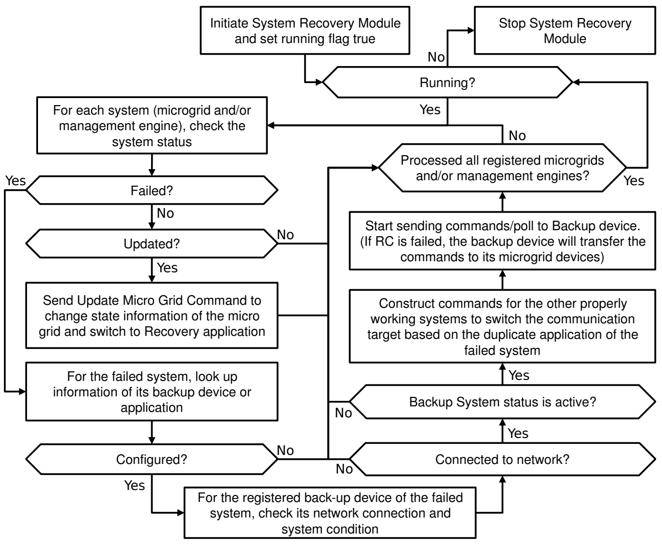
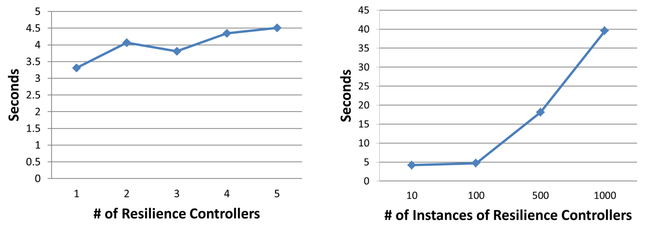

Initiate System Recovery Module and set running flag true
Stop System Recovery Module
No
Running?
Yes
For each system (microgrid and/or management engine), check the system status
No
Processed all registered microgrids and/or management engines?
Yes
Yes
Failed?
No
Updated?
No
Yes
Start sending commands/poll to Backup device. (If RC is failed, the backup device will transfer the commands to its microgrid devices)
Send Update Micro Grid Command to change state information of the micro grid and switch to Recovery application
Construct commands for the other properly working systems to switch the communication target based on the duplicate application of the failed system
Yes
For the failed system, look up information of its backup device or application
No
Backup System status is active?
Yes
Configured?
No
No
Connected to network?
Yes
For the registered back-up device of the failed system, check its network connection and system condition

Fig. 6. Procedure of System Upgrade and Recovery.



(a) Using actual servers in NEC Intranet.

(b) Created RC Instances within a server.

Fig. 7. Time of delay in switching from normal operation to back-up mode operation for modification of a management engine.

cost of setting up their own servers and easily enhance the memory space, computation resources, and communications speed with its virtualization mechanisms. EMS Storage is accessed by Operational Platform and Management Engines via Java Persistence API (JPA). Operational Platform and Management Engines communicate with each other using HTTP PUSH. As RCs are installed behind the company intranet, they use HTTP PULL to send and receive data from OP and MEs because of security issues. We employ HTTP Post and its message format is XML using RESTful Web Services, although the communications modules of each EMS support both HTTP GET and POST. Communications are secured using SSL and fingerprint authorization shared with both OP and RC.

Our focus in the validation experiment is on availability and resiliency when the system modification (e.g. Upgrade, Maintenance, Repair) and failure recovery procedures are conducted. The polling interval is set to 4 seconds. The test is repeated 10 times for each case and its average value is plotted.

A. System Modification of Management Engine

When system components need to be modified, upgraded, and/or repaired, availability is taken into account as a measure for fault tolerance and healing. The availability is assessed based on the following formula:

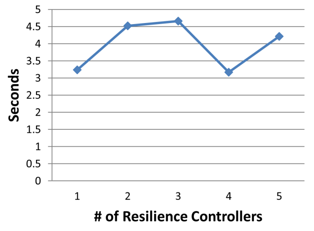
$$\text{Availability} = \frac{\text{Time Operating ``Normally''}}{\text{``Normal'' Operation + Off-line Time}}. \quad (1)$$

If the system modification start time is notified to the resilience controllers beforehand, the whole systems would maintain the normal operation just by switching the communicating management engine to the back-up module. However, if immediate attention is required to modify the primary management engine, we should send the system update signal as soon as possible to use the back-up module. Therefore, as to the validation experiment, it is important to minimize the off-line time so that when we could minimize the time that operational platform sends the signals to resilience controllers and receives the confirmation signals from management engine notifying that all the resilience controllers successfully switch the ME target to its back-up system, which leads to maximizing the availability. Figure 7 shows the time of delay in switching from the normal operation using a primary management engine to the back-up mode operation using secondary management engine, which is a duplicate module of the primary management engine. We first set up five servers for resilience controllers in NEC Intranet to validate the system modification behavior and its processing and communications time as in Figure 7(a). With slight increase in proportion to the number of resilience controllers, recovery time converge around 4 seconds, which is the polling interval from resilience controllers. Next experiment is to validate the behavior when operational platform and management engine needs to handle a number of HTTP requests to handle the system modification, which can be simulated by setting up RC instances within a server at NEC Intranet. Figure 7(b) shows its results based on the number of RC instances. Because of the large communications load on OP and ME, those communications modules take more time in processing a number of messages together with communications delay caused between AWS and NEC Intranet, although the system modification process is successfully validated even if the number of RC instances increases. An effective load balancing scheme on OP and ME needs to be addressed to handle huge number of RC servers while considering to increase computational resource from AWS.

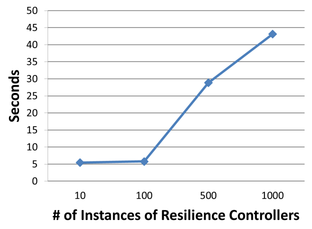
B. Failure Detection and Recovery

The definition of resilience is the ability to prepare for and adapt to changing conditions and withstand and recover rapidly from disruptions. Hence, a measure of resilience would consider "up times" to measure withstanding capabilities influenced by fault tolerance attributes and rapid recovery characteristics dependent on healing properties. Thus, resilience can be measured as

$$\text{Resiliency} = \frac{\text{Up Time}}{\text{Up Time + Down Time}}. \quad (2)$$



(a) Using actual servers in NEC Intranet.



(b) Created RC Instances within a server.

Fig. 8. Recovery time from failure detection to switching from the primary management engine to its back-up engine.

From (2), it is clear that we should validate the autonomous mechanisms to recover failure by minimizing the down time. We focus on the management system failure scenario again to validate the autonomous mechanisms. Figure 8 describes the downtime from the time failure happens to the time the failure is recovered by automatically detecting it and switching the management engine to back-up module. Although the validation results are similar to those of system modification experiments, the recovery time is slightly more than that of system modification as resilience controllers first need to detect the communications failure and switch to the back-up module autonomously.

VII. Conclusions and Future Work

We have presented an EMS architecture for resilient microgrid operations and autonomous mechanisms to dynamically configure and maintain microgrids as well as upgrade and recover from a failure. The EMS functionality is dispersed into Operational Platform and Management Engine utilizing EMS database storage that are all installed in public cloud platforms to save setup cost of IT platform and utilize their resources, while Resilience Controllers are deployed within our local intranet environment where microgrids and their devices are installed. Advisory functions are implemented in Management Engine so that it is able to aggregate the distributed resources and batteries to dispatch economic optimization commands to resilience controllers. Real-Time layer is in Resilience Controller as RC can communicate with local devices directly via local network protocols and collect measurement data constantly. An EMS Operational Platform is introduced as a system of modules that handle critical operational aspects of dynamic configuration and system recovery in order to minimize the cost caused from system interruption or outlier behavior of microgrids and devices. The autonomous functions are validated setting up the EMS environment in both AWS and NEC intranet and experimental results on system recovery time show the parallel nature of resilience controllers that automatically detect device failure independently and communicate with the back-up engine to sustain the EMS operation with minimum interruption, which reduces the cost of maintenance.

The presented validation focuses on the communications aspect with system recovering time to briefly assess the resiliency. We will need further analysis and experiments on the following aspects.

- How the communications failure and/or delay affects microgrid battery operations, frequency and voltage regulations, etc.;
- Load balancing on Management(s) Engine when handling a large number of microgrids;
- Distributed data duplication in case that EMS database crashes;
- Correlation among microgrids to extract their topological structure using graph modeling and dig into how a microgrid operation affects the neighboring grids.

References

[1] gtmresearch, "Grid edge quarterly executive briefing q1 2016," *https://www.greentechmedia.com/research*, 2016.

[2] A. Hooshmand, B. Asghari, and R. K. Sharma, "Experimental demonstration of a tiered power management system for economic operation of grid-tied microgrids," *IEEE Transactions on Sustainable Energy*, vol. 5, pp. 1319–1327, Oct 2014.

[3] A. Hooshmand, B. Asghari, and R. Sharma, "Efficiency-driven control of dispatchable sources and storage units in hybrid energy systems," in *2014 American Control Conference*, pp. 1686–1691, June 2014.

[4] B. Asghari, R. Patil, D. Shi, and R. Sharma, "A mixed-mode management system for grid scale energy storage units," in *2015 IEEE 6th International Conference on Smart Grid Communications (SmartGridComm)*, pp. 533–538, 2015.

[5] L. Mariam, M. Basu, and M. F. Conlon, "A review of existing microgrid architectures," *Journal of Engineering*, 2013.

[6] J. Kramer and J. Magee, "Dynamic configuration for distributed systems," *IEEE Transactions on Software Engineering*, vol. SE-11, pp. 424–436, April 1985.

[7] J. Schroeter, P. Mucha, M. Muth, K. Jugel, and M. Lochau, "Dynamic configuration management of cloud-based applications," in *Proceedings of the 16th ACM International Software Product Line Conference - Volume 2*, SPLC '12, pp. 171–178, 2012.

[8] J. Cao, A. Chan, Y. Sun, and K. Zhang, "Dynamic configuration management in a graph-oriented distributed programming environment," *Science of Computer Programming*, vol. 48, no. 1, pp. 43 – 65, 2003.

[9] S. Rivera, A. M. Farid, and K. Youcef-Toumi, "A multi-agent system coordination approach for resilient self-healing operation of multiple microgrids," *Industrial Agents: Emerging Applications of Software Agents in Industry. Berlin, Heidelberg: Springer Berlin Heidelberg*, pp. 1–20, 2014.

[10] S. Rivera, A. Farid, and K. Youcef-Toumi, "Coordination and control of multiple microgrids using multi-agent systems," *Energypath 2013: Our Global Sustainable Energy Future*, pp. 1–5, 2013.

[11] R. T. and A. S., "A cloud computing approach for power management of microgrids," in *IEEE PES Innovative Smart Grid Technologies - India (ISGT India)*, pp. 49–52, Dec 2011.

[12] M. Parandehgheibi, T. Konstantin, and E. Modiano, "Modeling the impact of communication loss on power grid under emergency control," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2015.

[13] M. Parandehgheibi, E. Modiano, and D. Hay, "Mitigating cascading failures in interdependent power grids and communication networks," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp. 242–247, 2014.

[14] M. Parandehgheibi and E. Modiano, "Robustness of interdependent networks: The case of communication networks and the power grid," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 2164–2169, 2013.

[15] J. Rocabert, A. Luna, F. Blaabjerg, and P. Rodrguez, "Control of power converters in ac microgrids," *IEEE Transactions on Power Electronics*, vol. 27, pp. 4734–4749, Nov 2012.

[16] O. Alliance, "Openadr 2.0 b profile specification," *Raportti. OpenADR Alliance*, 2013.

[17] P. Felber, X. Defago, R. Guerraoui, and P. Oser, "Failure detectors as first class objects," in *Proceedings of the International Symposium on Distributed Objects and Applications*, pp. 132–141, 1999.