SAND REPORT

SAND2005-7937
Unlimited Release
Printed January 2006

Agent-Based Control of Distributed Infrastructure Resources

Laurence Phillips, Hamilton Link, Randy Smith, and Laura Weiland

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2005-7937
Unlimited Release
Printed January 2006

Agent-Based Control of Distributed Infracstructure Resources

Laurence R. Phillips
Critical Infrastructure Systems Department

Hamilton Link and Randall Smith
Advanced Information Architectures Department

Laura Weiland
System Sustainment and Readiness Technologies Department

Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-1368

Abstract

Managing distributed infrastructure resources is usually accomplished by telephone calls among the managers and operators. This works reasonably well under ordinary circumstances but breaks down—often catastrophically—under stress. Individual motivation, long response times, and poor situation awareness interfere with operation and can even cause breakdowns. Broadly distributed operations would more robust and fail more gracefully than centralized systems, but remain unlikely given the difficulty in operating large infrastructures even with modern Supervisory Control And Data Acquisition (SCADA) systems. The solution is management by distributed software that maintains normal operation, enforces operational and security policy, deals with contingencies, and protects against malicious indsiders, errors, and outright attacks. We specify a distributed agent coalition able to accomplish this for distributed electric power and describe a prototype implementation based on Sandia-developed technology.

— This page intentionally left blank —

Table of Contents

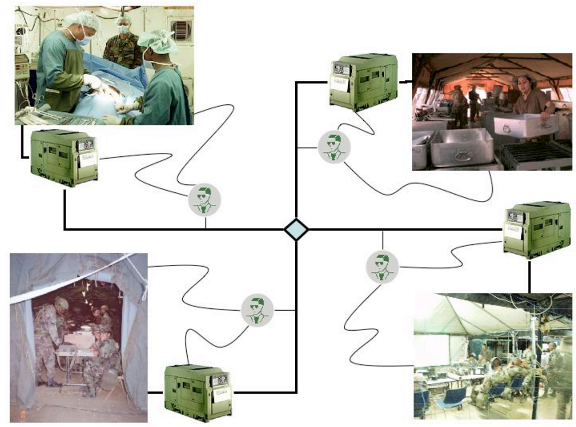
Figures

Tables

Executive Summary

In this report we examine the application of agent technology to the management and control of electric power microgrids. Our contribution has been to define, and demonstrate elements of, a secure extensible framework for agent-based management of distributed resources that supports the necessary capabilities.

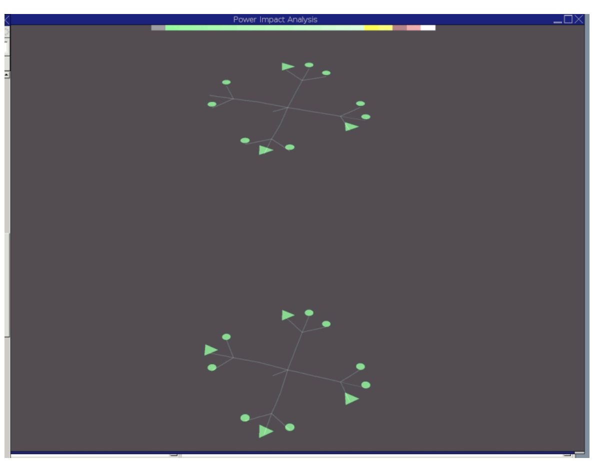
The essential thesis of this project is that agents can operate utility networks and that a network operated by agents is more robust and reliable than a network operated by conventional means. Although many have pointed out that agents have the correct attributes—autonomy, speed, repeatability, intelligence—for managing distributed resources, in particular electric power networks, we found the functional descriptions of such agents in the literature lacking in terms of saying what they actually *do*.



Agents cooperate with one another to manage a military camp microgrid

This report is our response to that lack. The most pertinent “missing ingredient” that agents can supply is *management* capability: Situation awareness and responsiveness to observed conditions that would enable (nearly) autonomous automated software to operate physically separated power sources as if they were co-located.

In theory, microgrids are robust and economical: To negate an equal amount of production, more failures (albeit smaller ones) are required for a network of many small sources compared to one with only a few large power sources. Power transmission distances are smaller and co-produced heat can be utilized more readily. Our decision to study microgrids was prompted by DOE’s desire to increase the production of electricity by “micro” (< 200kW) power sources. Replacing one percent of the nation’s existing generators with microsources would require 48,000 new installed 200kW generators, quadrupling the total number of generating units. This doesn’t prove agents are required to operate a microsource-based power grid, but it does imply that deployment can occur only if some managing/operating mechanism can perform day-to-day operations and respond to at least simple contingencies. We contend that agents have that potential.



Visualization of the military camp microgrid before (bottom) and after halting the leftmost generator

We have undertaken to design a distributed agent-based system that enables elements of a microgrid to cooperate not only with one another, but also with other agents operating other microgrids. The agents sense the states of power system components and make collective decisions about how to respond to user requests, expected events, and unplanned contingencies. The ABCDIR agent system manages electric power generation and distribution by monitoring voltages and power flow, accepting requests from human operators, and recognizing and responding to conditions of interest. It takes action when the current state indicates a reconfiguration of the power components is necessary to achieve liveness conditions or avoid violation of safety conditions.

It's easy to adopt a straightforward stance towards managing infrastructure—"do the correct things in the proper order"—but this is difficult if the system is complicated, as such systems tend to be. Operators are hampered by poor situation awareness, by lack of information about likely future system state, and by a general inability to agree on what system state they'll cooperate to achieve or to construct a series of steps likely to achieve it. It becomes nearly impossible if they must communicate by telephone. Providing a relevant answer is a complicated process; status-containing data structures are complex; and maintaining a coherent, common system state picture using multiple data sources is difficult.

For this project we defined a "military camp" setting—four tents: HQ, barracks, mess, and hospital—for use in describing how the agents should react to various situations. The microgrid consists of four generators (one per tent), each paired with a critical load and a non-critical load. Each load operates independently of the other loads. A communal bus connects the four subsidiary power systems. This enables power from any generator to be used at any load. We constructed a simulation of this microgrid for manipulation by the project's agents.

The agent system for operating the four-tent microgrid consists of agent classes that define the necessary policies, object models, roles and interactions among roles, protocols, liveness and safety conditions, messages to be sent and received, and appropriate responses for agents operating the microgrid. When conditions arise that imply a response by the agent system, the agents recognize this and execute various multiparty protocols as dictated by policy to arrive at a state of consensus about what, if anything, to do. Possible behaviors—some by individuals, and some by the entire group—include planning, plan execution and monitoring, prediction, state estimation, resource dispatch, fault isolation and recovery, notification, and human interaction. We do not address issues of commerce and power storage in this report.

We implemented a small demonstration system to provide proof of principle and to show that a complete implementation of the system design that resulted from our analysis could be accomplished. The demo shows agents interacting with a human operator to respond to a request to shut down one of the four camp generators.

Although much work remains before coalitions of autonomous agents should be allowed to manage real-world power systems, we believe the principles, framework, and framework extensions described in this report lay effective groundwork for this task.

1. Introduction

In this report we examine the application of agent technology to the management and control of electric power microgrids[1]. Our thesis: Agents are a valuable framework for operating microgrids. Intelligent agents offer a secure, distributed means to manage electric power production, enforce system policy, and respond to unexpected events in the power network.

Many have pointed out that agents have the correct attributes[2]—autonomy, speed, repeatability, intelligence—for managing distributed resources, in particular electric power networks. Agents have been demonstrated carrying out important elements of the needed functionality, and many algorithms suitable for use by agents have been described. See Section 2, "Related Work," for further discussion. Our contribution has been to define, and demonstrate elements of, a secure extensible framework for agent-based management of distributed resources that supports the necessary capabilities.

In theory, microgrids are robust and economical: To negate an equal amount of production, more (albeit smaller) attacks are required against a network of many small sources compared to one with only a few large power sources. Transmission losses are lower for smaller sources operated near the loads they serve. Cogenerated heat, often discarded as waste, is easier to use in a microgrid because thermal loads, if any, are likely to be near the electrical loads, which, in turn, are likely to be located near the power sources.

Our decision to study electric power was prompted by DOE's desire to increase the production of electricity by "micro" (< 200kW) power sources. This cannot happen to any substantial degree unless a suitable automated management system can be identified. In our original proposal for this work, we said: "Widespread penetration [of the electric power network by microsources] can occur only with automated control because the large number of elements precludes manual control."

According to DOE's Energy Information Administration, over 16,000 existing U.S. sources produce an average of 60 MW apiece (source: [EIA04]). Replacing one percent of these with microsources would require 48,000 new installed 200kW generators, quadrupling the number of generating units. This doesn't prove agents are required to operate a microsource-based power grid, but it does imply that deployment can occur only if some managing/operating mechanism can perform the bulk of day-to-day operational tasks and respond to at least simple contingencies. We contend that agents have that potential.

When we proposed this work—2 1/2 years ago as we write this report—there were few, if any, operational microgrids. [Lasseter02] states that microgrids as envisioned in 2001 "represent an entirely new approach to integrating DER." The situation has not changed appreciably over the duration of this project.

In May 2001, the U.S. Department of Energy's (DOE) Transmission Reliability Multi-Year Program [DOE1] called for demonstration of a microgrid by 2005. In June of 2005, a DOE

---

[1] The DOE defines a *microgrid* as "at least one distributed resource that is capable of operating either in parallel with or independent from an electric power system while providing continuous power to multiple loads on the electric power system." A microgrid is capable of operating standalone; i.e., of supplying the expected loads while not connected to a larger grid.

[2] Not all agents have all the attributes, but various instantiations of agents have displayed all the desired traits at one time or another.

call for proposals [DOE2] in the area of power transmission and distribution stated: "Further development, testing, and demonstration of microgrid designs are needed to better understand their operations and integration with electric power systems" and "the potential value of DER[3] to support distribution operations/automation is under-realized." The 2005 call goes on to say that "even more significantly, customer loads, generation, and storage are not managed and operated to benefit the distribution grid." The requested deliverable is the demonstration of an operational microgrid. This call implied that DOE had not seen a satisfactory microgrid as of June 2005, nearly four years after having called for them.

Nevertheless, as this is being written, many small sources are operational, ordinarily as ancillary power sources for peak-shaving and emergency power. There are also many examples of multiple co-located small sources operating in concert to meet larger needs on an ongoing basis. Most are configured as "mini power plants": several co-located identical sources connected to one bus and operated by a single cadre of engineers. Neither of these situations is a microgrid: multiple cooperating small sources that are *not* co-located. When multiple sources are operated at physically separated sites, they are almost always operated independently, satisfying local loads when needed and otherwise shut down.

Operating in conjunction with the main grid is less difficult than "islanded" operation. Once power coupling issues—non-trivial but straightforward—have been addressed at a given site, power needs are met by the main grid and excess power is absorbed by the main grid. In a microgrid, however, power must be balanced over a playing field much smaller than the regional grid, with limited options and storage capacity. Operators must be ready to respond to increases in load that exceed the capacity of operating sources. The ability of operators to maintain situational awareness and to respond quickly and correctly diminishes as the sources get farther apart and communication among the operators becomes difficult.

Distributed resources are difficult to operate because it's hard to get coherent status information and to coordinate local actions to meet global requirements. This is required because decisions about how to operate the components of a distributed system cannot be made independently of one another. A distributed system is still a system, i.e., "interrelated interacting artifacts designed to work as a coherent entity [WordNet]".

In pipeline systems (for managing, e.g., chemicals, natural gas, petroleum, water, sewage), command inputs operate valves and pumps. Poor coordination can result in cross-contamination, accidental release, pressure overages, infrastructure damage, and injury or death to human operators. Review of the U.S. Chemical Safety and Hazard Investigation Board's completed investigations [CSB] shows several recent fatal accidents in these categories. [NTSB1] and [NTSB2] discuss fatal accidents due to natural gas pipeline failure. [Safire] speculates that adversarial penetration of a natural gas pipeline's SCADA system was used to cause system failures leading to an explosion visible from orbit.

In electric power, command inputs operate switches and generators. Poor coordination can result in inefficient operation (generators operated at lower or higher levels than optimum; unnecessary power transmission and generation; wasted co-produced heat), unintended activation, unintended shutdown, infrastructure damage, and injury or death from

[3] Distributed Energy Resources (DER) are small (<200kw) power sources, storage systems and demand response elements linked to the main power grid.

accidentally energized circuits or accidental equipment startup. It is generally agreed that poor situation awareness and slow, uncoordinated response to recognized faults were instrumental in the August 2003 Northeast Blackout [NERC].

It's easy to adopt a straightforward stance towards managing infrastructure—"do the correct things in the proper order"—but this is difficult if the system is complicated, as such systems tend to be (see Figure 1, a depiction of the relatively simple IEEE standard 118-bus system). Operators will be hampered by poor situation awareness, by lack of information about likely future system state, and by the general inability of any group of people—even if they're all in one room—to agree on a system state they'll cooperate to achieve (Given that a contingency has occurred, what should be the state of the 118-bus system?) or to construct a series of steps likely to achieve that state. It becomes nearly impossible if they must communicate by telephone. Our point here is that providing a relevant answer is a complicated process, status-containing data structures are complex, and maintaining a coherent, common system state picture using multiple data sources is difficult.

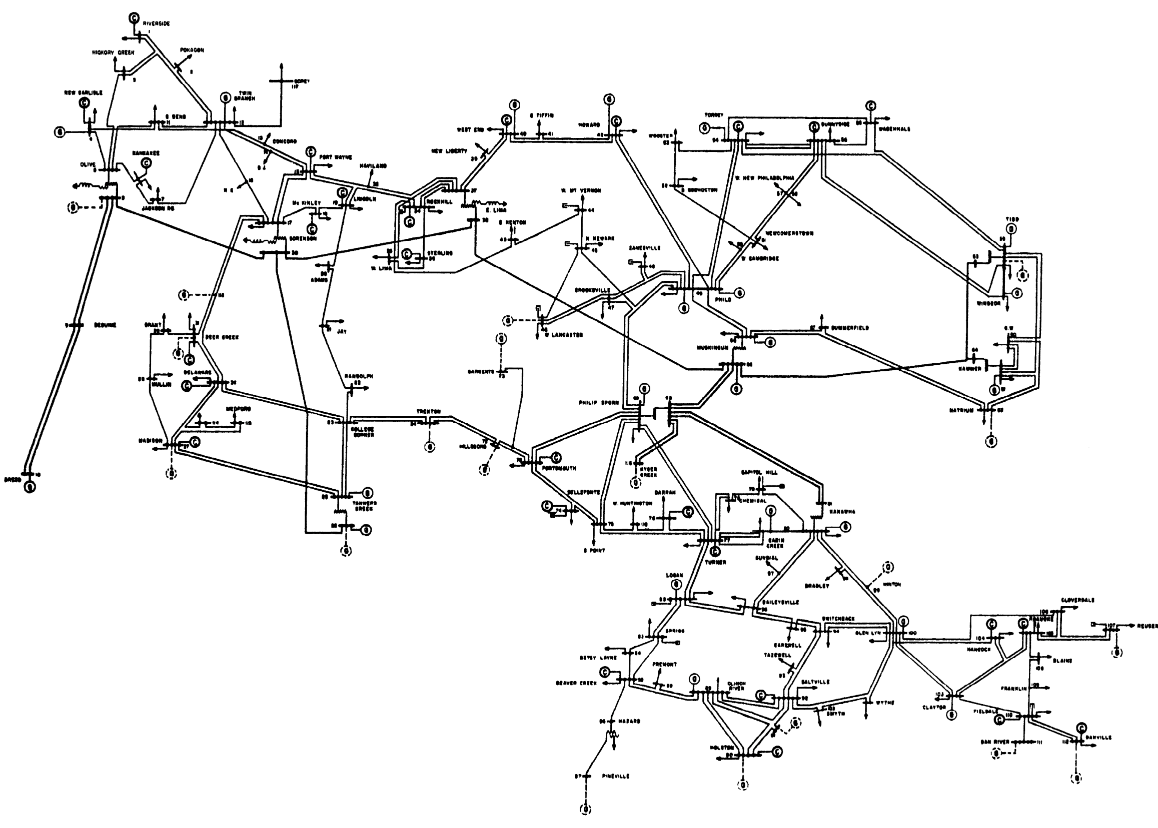


Figure 1: The relatively simple IEEE standard 118-bus system

We recognized in the early stages of this project that our contribution would not involve classical control, although control technology is fundamental in the operation of electric power systems. A distributed agent coalition almost certainly needs several seconds to organize, plan, and execute coordinated activity. Though much faster than humans at such tasks, it is woefully inadequate to coordinate activity at the microsecond time scales needed for control and stabilization of the A/C waveform.

We expended significant mental activity discovering an appropriate role for agents. In time we realized that the most pertinent "missing ingredient" that agents can supply is *management* capability: situation awareness and responsiveness to observed conditions that would enable (nearly) autonomous automated software to operate physically separated power sources as if they were co-located. The project would have been more accurately titled: "Agent-Based *Management* of Distributed Infrastructure Resources," but the original title—abbreviated "ABCDIR"—stuck.

The ABCDIR agent system manages the utilization of electric power generation and distribution components. It does this by monitoring voltages and power flow at strategic locations, accepting requests from human operators, and recognizing and responding to conditions of interest. It takes action when the current state indicates reconfiguration of system components is necessary to achieve liveness conditions or avoid violation of safety conditions. This requires on the order of tens of seconds to several minutes.

In steady state, agents periodically sample local sensor data to populate the local elements of its internal model of the grid. When significant changes in local state occur, the responsible agent shares this new state information with the other agents. "Significant" means that voltage or power magnitudes have changed beyond a threshold value for a duration that exceeds a threshold time. It may be necessary to tune these thresholds via a learning mechanism. A threshold trigger initiates action planning, e.g., "The power flow through *circuit_breaker_X* indicates that *distribution_line_X1* has been running at 93% of capacity for 38 seconds. If this condition persists for 6.2 minutes, the breaker will open to prevent heat-induced line sag. Initiate a planning cycle to reduce the power flow to 80% or less."

Consequence assessment and action planning can also be activated by operators, e.g.: "At 1106 hours on 9-12-05 authorized operator *JohnHawkins* requested that *Generator#549* be taken off-line." Policy elements held by the agent can be arranged to permit appropriately certified and authorized entities to override reluctance on the part of an agent coalition to proceed in the face of negative consequences it predicts.

The remainder of the paper is organized as follows: Section 2 (p. 14) discusses related work. Section 3 (p. 19) describes our approach and explains why we believe it is appropriate. Section 4 (p. 20) expresses our vision and explains the system design. Section 5 (p. 26) examines microgrid deployment. Section 6 (p. 31) describes the use case we developed to illustrate the properties of the agent system. Section 7 (p. 35) describes the microgrid modeling effort needed to support the agent system demonstration. Section 8 (p. 55) lays out the ABCDIR agent system design. Section 9 (p.59) describes the interactions among the agent roles. Section 10 (p. 66) details the protocols executed by the agents in operation. Section 11 (p. 75) describes the way the agents are organized. Section 12 (p. 79) outlines the classes of agents needed. Section 13 (p. 81) describes an instantiated agent and how it may take on multiple roles. Section 14 (p. 82) describes the agent communication network. Section 15 (p. 84) discusses the implementation process. Section 16 (p. 97) describes the demonstration and some aspects of the implementation framework. Section 17107 (p. 105) states our conclusions and lessons learned. Section 18 (p. 108) outlines future work. Section 19 (p. 110) lists references and sources. Appendix 1 is a short glossary of terms used in the report and Appendix 2 gives the notation for liveness expressions.

2. Related work

Work in the area of agent-based infrastructure management can be organized into two general categories: *Operation* and *Fault Recovery*. Published work concerning operation tends to be at a high level, consisting in large part of position and strategy papers.

One exception deserves further consideration: [Rehtanz] is an extensive examination of the roles agents can play in power system control and operation. The book contains material by 26 internationally located contributors. Most selections include some discussion of application and implementation. Some include case studies. Work pursuing the application of agents to management and operation of power systems should find this resource valuable.

Published fault recovery work generally presents algorithms appropriate for execution by a distributed system, nominally an agent coalition.

Each category can be further broken down into work that involves agents and work that does not. For work not involving agents, it is generally of interest to determine whether the presented work is suitable for agent execution. The literature concerning operation of electric power systems is extensive and we do not attempt to do more than indicate a small selection we found particularly interesting.

The area of distributed systems operation is of particular interest, however, especially system state estimation, fault detection and recovery, and distributed operation algorithms, since relevant protocols and algorithms can be implemented directly in distributed agents.

2.1. Related work in agent-based infrastructure operation

In this area, comparing our work with the work of others helped shape our design as we noted differences in organization, philosophy, and technology. There are two primary distinctions between our work and what we observe in the literature: First, published work tends to be high-level, with the benefits of agents being matched against the needs of distributed operations, but only marginal information, if any, concerning reduction to practice ([Rehtanz], again, is an exception). Second, published work in this area primarily presents existing agent frameworks as execution mechanisms to achieve power system function, in many cases only tangentially addressing the requirements of the agent framework itself.

[Amin99] argues that agents have the appropriate capabilities to operate the national power grid, but doesn't explain their function as they accomplish the task. [Amin01] presents a fairly detailed view of a multi-layered, multi-species multiagent system that includes, among others, fault isolation agents, command interpretation agents, and event identification agents.

[Arnheiter] describes a market-based approach for allocating electric and heating energy. Agents assigned to each household and power plant act independently to and improve real-time performance by reducing data transmission volume. Simulation results show high-quality load management and efficient power plant operation.

We agree with [Brazier] that "dynamic load management of the power grid is essential to make better and more cost-effective use of electricity production capabilities." The paper gives a revealing but abbreviated view of full agent tasking and describes a multi-agent system capable of negotiation for load management. The ability of multiple agents acting on behalf of the utility and its customers to agree on a unit price for power is demonstrated.

[deAzevedo] points out that agent technology is a promising approach to support the construction of a "new generation" of energy management systems (EMS) and describes an EMS agent framework. This is a position paper that concludes, "The controlled, organized, and autonomous architecture of today's centers will become open and dynamic. Furthermore, the connection with other players will be of the utmost importance. This paper shows that the use of agents is an effective alternative to treat the uncertainties of that new scenario."

[Dimeas04] and [Dimeas05] together present a relatively complete picture of agents operating a microgrid. These papers recognize and address the impossibility of centralized control among several owners and present the operation of a multi-agent system that uses agents advantageously to execute a classical distributed algorithm. The agents are developed on the JADE (Java Agent DEvelopment) framework and interact with a small operating microgrid (one rotating mass generator, one photovoltaic generator, batteries, load) connected to the main power grid. These papers focus on agents buying and selling electricity. We would utilize the work presented here to improve our agents' ability to participate in an energy market, which we have not pursued significantly, although we recognize the requirement to do so.

[Petrov] shows how independent agents can act in a predatory manner to deny suppliers of generated power access to customers and exploit vulnerabilities to achieve market destabilization. The authors point out that "reducing the power delivered to a customer is sometimes the best solution to the total congestion management optimization problem" caused by such predatory market activities. The paper considers a small power network where "an independent entity takes care of the congestion management as well of allocation of the available resources. One of the companies engages in predatory behavior, using the congestion management policies combined with carefully chosen bids to cut off one or more of the generators of the other company. Vulnerabilities associated with shutdown and startup costs, minimum up and downtimes, [and] ramp rate and generator limits for each generator are utilized to achieve market destabilization." The paper makes a convincing case that self-interested agents can cause problems in an open environment by gaming the regulatory policies, even though following them to the letter.

[Tolbert] presents a scalable multi-agent paradigm to control distributed resources with the intent of achieving higher reliability, better power quality, and more efficient generation and consumption. The paper describes general capabilities and kinds of knowledge needed for power management tasks, and describes mathematically two examples of agent function, one for injecting reactive power to improve stability and another for minimizing non-active power. The paper does not discuss agent frameworks, cooperative behavior, or security. The mathematical content is directly extensible to agents in an operational system.

[Thorp] presents simulation results of "protection agents" engaged in realistic protection and control scenarios. The agents increase system protection performance by exchanging basic information, primarily contributing to fault identification and isolation. We agree that "communication security, proper communication protocol selection, and the appropriate range of applications for agent-based methods *[All of which we discuss in this report – LRP]* are just a few of many issues that must be dealt with before agents will be ready for use in real systems."

2.2. Related work in agent-based fault recovery

Fault recovery is of particular interest because it demands dynamic operation of the power management system. Our efforts focused on the top of the requirements pyramid for a distributed agent-based system able to carry out fault recovery operations. Related work in this area contributed to our project by allowing us to determine whether the agent framework in this report could support the desired functionality we observed in the literature. The overt difficulty we perceive is that much of the published work discusses only fragments of what's needed. We recognize that this is the nature of the scientific endeavor: Knowledge is not gained all at once, but over time, as hypotheses are proposed and examined. Although we consider this an important topic, integration of a suite of published fault-isolation and -recovery processes into our framework is currently an element of future work. Work described in this section is a representative sample of what we would include in this effort.

[Wang] proposes secondary voltage control based on the principles of multi-agent system theory and presents an example power system that illustrates successful multi-agent coordination using secondary voltage control in system contingencies. Secondary voltage control has primarily been used to improve power system voltage stability. In this paper, it is used in a new way: voltage management during system contingencies. The secondary voltage control implementation discussed here is based on the principles of multi-agent system theory. An example power system is presented to demonstrate the necessity of secondary voltage control among an AVR, an SVC and a STATCOM installed in the power system, and also to illustrate the success of applying the multi-agent co-ordination of secondary voltage control in system contingencies.

[McDermott] provides a reconfiguration algorithm for reconfiguring a power system after failures have occurred. The algorithm is essentially gradient ascent with backtracking. This method's main advantage is the accurate treatment of voltage and current constraints, including the effect of control action. An example shows how the algorithm reaches an optimal result while satisfying voltage constraints.

In comparison, [Bretas] uses artificial neural networks to restore and configure a power system that has been broken into islands. The islands are configured in parallel by neural nets, and as the islands are restored the tie lines connecting them to one another are closed. The neural net algorithm is impressively fast and provides good results. Two other useful algorithms are also presented. The first is a breadth-first search through all possible configurations to produce a functional power system restoration plan. This algorithm was used to create training examples for the neural net and for validation. The other is a switch-sequencing algorithm that takes into account load priority and the desired final configuration.

[Butler-Purry] provides a method for post-contingency shipboard service restoration, which is especially interesting in the context of microgrid operation. Fast restoration and load priority are taken into account. We were particularly intrigued by the characterization of load as "vital or non-vital," which resembles our "critical/non-critical" characterization. The method is illustrated with case studies of a simplified shipboard power system.

[Hotta] describes a real-time restoration expert system for a dispatching center. The system has two modes: The on-line guide mode, which provides a restoration plan and procedures using a system restoration knowledge base; and the off-line simulation mode, which can be used to verify the validity of acquired knowledge. Implementation issues and execution results from the guide mode are provided. It is particularly relevant to our work that the system is designed for use in real time. We are particularly intrigued by the notion that system restoration procedures have been captured in a rule set not unlike the liveness and safety conditions we produced during our analysis process[4]. This rule set would be valuable in designing an agent-based system for deployment.

2.3. Related work in distributed system operations

Any distributed algorithm (or non-distributed algorithm, for that matter) that provides improved metrics for a given task is potentially useful for that or similar tasks, regardless of whether it is implemented within an agent framework. Our primary research purpose was to explicate the nature of an agent framework useable for infrastructure control and secondarily to describe and possibly exhibit relevant algorithms. Although we did not focus on selecting from the literature, or otherwise discovering, the best algorithms for all the necessary functions, we nevertheless discovered significant pertinent material.

For example, [Phillips] describes a system of cooperating Flexible A/C Transmission Systems (FACTS) devices, discusses how agents would be utilized in that context, and describes several potential vulnerabilities and protective measures. The essential FACTS device greatly enhances the stability of the power flowing over the transmission line to which it is attached and can alter the effective impedance of the line, thus preventing line overload.

Several properly placed FACTS devices can cooperate to optimize the power flow of an entire transmission network. The pairing of agents and cooperating FACTS devices is natural because [Chaloupek] presents the results of using a genetic algorithm to determine optimal or near-optimal FACTS device placement. [Armbruster] describes a distributed maximum flow algorithm to optimize power delivery and [McMillin] describes a means by which a set of FACTS devices can itself recognize some faults and prevent others, including some adversarial intrusions. [Ryan] uses high-order object models to describe the operation of a system of cooperating FACTS devices. These papers and reports provide significant relevant information concerning power management using agents and FACTS devices.

[Rudnick] presents a methodology to reconfigure an electric power distribution network under normal operating conditions to reduce the active losses of the network or to balance the load of the system's feeders. The straightforward approach uses a heuristic algorithm that compares power losses among various branch configurations. The result is not guaranteed to be optimal, a common “feature” of heuristic algorithms, but the algorithm converges quickly and test results follow closely the exact results calculated for comparison. Algorithms of this type are valuable in real-time systems because the best result found up to that time can be returned at any moment. The algorithm presented is applicable to radial networks. We would expect several algorithms similar this one, relating to various system conditions and configurations, to be utilized by power management agents as the situation demanded.

[4] Compare, e.g., “If restoration without overload is possible with the source that had fed the current outage area before the fault, then that source is used” from [Hotta] with Section 8.3 of this report.

3. Approach & Rationale

The essential thesis of this project is that agents can operate utility networks and that a network operated by agents is more robust and reliable than a network operated by conventional means. Although this assessment is echoed in the literature, we found the functional descriptions of such agents somewhat lacking in terms of saying what the agents actually *do*. In [Amin01], for example, the function of some proposed agents is explained as follows: "The agents in the coordination layer continuously compare the world models between the deliberative and reactive layers. They update the current real-world model and check if the plans (or commands) from the deliberative layer represent the system's current status." This level of description is relevant and can serve as guidance—indeed it implied that we were on an appropriate course—but it is not a design.

We have undertaken to design a distributed agent-based system that enables elements of a microgrid to cooperate not only with one another, but also with other agents operating other microgrids. Stating the function of the agents in enough detail to serve as a design basis was adopted as the project goal and proved to be a very challenging aspect of designing an agent-based system to manage an electric power grid.

Fairly early in the project we realized that inclusion of the word *control* in the project title (and therefore the title of this document) could be misleading because it evokes—inaccurately, with respect to our work—a control-theoretic, mathematical approach to dynamical systems. While control theory is certainly relevant in this context, our motivation to use agents is not for electric power system *control*, in the sense that a speed governor controls an engine or a thermostat controls a furnace; this is already accomplished by "conventional" (non-agent-based) power control systems.

In distinction to control, agent involvement enables rapid, automated, distributed decisionmaking based on policy; in other words, distributed, coordinated *management*. The decisions made are, essentially: which resources to use to satisfy load requirements under a given set of conditions, what actions to take under various conditions of rapid change, and with whom and in what manner to interact in order to make these decisions with the appropriate authority.

Agent-enforced policy orientation gives human policymakers the capacity to define an explicit policy that the agents endeavor to enforce in the face of contingencies, threats, and unpredicted behavior by human operators. An agent of this sort acts either in direct response to an authorized operator (or group of operators) or because its internal data is consistent with behavioral premises that so acting will increase the likelihood of satisfying a liveness condition (or reduce the likelihood of violating a safety condition) and is consistent with policy.

We utilized two methodologies: *Use case analysis* to define the problem (see Section 6) and the *Gaia* methodology to design the solution (see Section 8). We refer to the management system and the agent framework we developed during this project by the acronym of the report title: Agent-Based Control of Distributed Infrastructure Resources (ABCDIR).

— This page intentionally left blank —

4. Vision and High-level Design

This section describes a system of agents that manage and control a microgrid. A microgrid is made up of electric power generators, electrical loads, and a distribution system that allows power to flow from the sources to the loads[5]. Breakers and switches govern the interconnection of these elements. Power is generated and distributed to satisfy demand. The combined loads served by a set of generators can never exceed the total generation capacity of those generators. In case of excess demand, power is allocated based on policy.

The agents sense the states of power system components and make collective decisions about how to respond to user requests, expected events, and unplanned contingencies. Each agent has both local and group responsibilities. Here, we consider a *closed* coalition of agents: the agents are based on the same framework and are therefore similar to one another; they are informed of one anothers' existence and expect one another to behave according to well-known strategies and policies. This does not preclude malicious co-option of agents by intruders; the agent framework incorporates technology designed to prevent or mitigate the effects of such takeover.

If agents like these are deployed in the context of other agents of different species that may also be operating power grid components, the system will be *open*: "our" agents—those whom we control—will interact with, and might form associations with, agents responsive to different policies and authorities with goals different from our own. Our agents will need to be more complicated: They will need to know how to negotiate with other self-interested entities and to deal with broken contracts and agents operating in bad faith. In a closed system, agents designed and given policy by us interface only with one another; in an open system our agents would have to deal with agents of other species serving other masters[6].

4.1. Organizing Principles for Agents Managing a Network

Cells, globs, and cooperatives are the organizing and scaling concepts for agents operating a distributed system. A *cell* is a set of sources, loads[7], switches, branches, and buses managed by a single agent. The idea behind a cell is to attach an ontological element—the cell—to the smallest unit of agent responsibility. Cell extent is limited so that operating a cell under ordinary conditions can be presumed to be straightforward, consisting essentially of maintaining adequate power for the loads that "belong" to the cell. A cell should be conceptually simple and arranged so a single "tactical monitoring and control" agent could manage it easily; e.g., with sources physically collocated and all on one bus and with loads switched *en masse* as either "critical" or "non-critical".

When two or more cells are electrically connected to one another, the possibility of trading power among cells arises. Cell interaction policy is the locus for the economic rules that

---

[5] Power systems often include significant storage capacity, especially small systems designed to serve local loads. We have not included storage in our analysis, but we would expect to do so in follow-on efforts, especially if considering deployed or deployable systems.

[6] We used the analysis process specified in [Zambonelli] to develop specifications for the ABCDIR agents. One valuable aspect of this reference is that it includes two running examples for each step of the process, one for a closed agent system, the other for an open agent system.

[7] Loads are not customarily associated with sources, but they would be in this formulation by virtue of being assigned to the agent that's managing the sources.

govern power transactions among cells and determines the nature of the organizations the cells can form. A group of connected cells can be organized in two ways: a glob or a co-op. If generators are physically separated from one another to the extent that they can't readily be operated by a single human technician, and they aren't on the same bus, meaning their power is distributed via a network of switches, breakers, buses, and lines and can be routed or islanded, those generators are better characterized as a group of separate cells, either a glob or a co-op, depending on the role of policy in their operation. Globs and co-ops look alike, but the agents that manage the components base their behavior on different kinds of policies.

A *glob* is a network of cells in which the member cells have agreed to trade power. A glob differs from an unconstrained collection of connected cells in that the cells in a glob need to be able to reason about ontological elements that relate to power trading (kilowatt-hour volumes, tariffs, payments, prices, contracts, etc.). Agents without these elements could not technically form a glob because they lack the knowledge to conduct power business. The nominal behavior of a glob-capable cell is to produce *or acquire* adequate power for cell loads. Basic standalone cell policy would be supplanted by liveness and safety conditions specifying how and when to trade ("buy power if it's cheaper than it costs you to produce it yourself," "buy power if you can't make enough," etc.). Glob membership is attractive to a glob-capable cell agent because it can import available power from other cells if in-cell loads can't be satisfied (or can't be economically satisfied) by in-cell sources. A cell agent in a glob would "prefer" to purchase power rather than not satisfying its loads and to spend less on purchased power than on power it makes with its own sources.

A *co-op* is a glob in which the cells obey a common policy governing transactions among the constituents. In particular, policy may dictate that a cell give preferential treatment to loads not its own. Co-op cells would need all the capabilities of glob-capable cells. The primary distinction between a glob and a co-op is that co-op policy is designed to affect power over the entire group of cells—to maximize the probability that critical co-op loads will be satisfied, for example—while there's no such overarching design in a glob. A cell in a glob satisfies its own loads before considering other cells, whereas a cell in a co-op *may*, based on policy, satisfy loads in other cells before it satisfies its own. A glob becomes a co-op when its cells begin to obey a policy that overrides individual cell policies; e.g., "a cell in a co-op shall shed its own non-critical loads when necessary to serve another cell's critical loads."

4.2. Behavior of the Organizational Elements

The default behavior of an agent managing a cell is *greedy*: it acts to supply in-cell loads. If the sources in a cell can generate more power than required by in-cell loads, a cell agent may export power if it is part of a glob. For the moment, we assume price conditions are always met; i.e., there might not be enough power from in-cell sources to satisfy in-cell load, but this would not be because of economic reasons. This raises the prospect of a cell/co-op-based power economy that admits auctions, etc., which we leave for future consideration. Note that has no bearing over supply decisions within a cell, because power is always applied preferentially to in-cell loads in the canonical cell. This would not prevent the appropriate tariffs from being collected nor free the agent from having to know about them.

The default behavior of cells in a glob is unspecified and depends primarily on the commitments, if any, of its constituent cells. Interconnections between cells from two different globs just makes a bigger glob; there's no notion of two globs of cells interacting as globs[8], because globs don't have an identity, nor is there any overarching mechanism to differentiate a cell as belonging to a particular glob. Emergent or unstable conditions might result from cells responding to loads in other cells as specified by their individual contracts.

The glob organizing principal exists primarily to describe "life beyond the cell" so that co-ops can be discussed. Co-ops make sense when some loads are more important than others; greedy cells want to satisfy their loads, and if all loads are alike, it doesn't matter which ones get satisfied first. But in a glob, even if some loads are more important, unimportant loads might be satisfied while important loads nearby remain unsatisfied (because, say, generators have failed in that cell) because of default greedy cell behavior.

The default behavior of cells in a co-op is defined by the policy of the co-op and can be made to appear altruistic; for example, policy might dictate that a hospital without power be supplied by its neighboring cells, even if they must shed advertising and entertainment loads to do so. Exactly what loads are shed under what conditions is determined by co-op policy and enforced by the agents.

We suggest that a co-op should be, without loss of generality, either an actual multi-component co-op or a primitive cell following co-op policy. This is conceptually appealing because if a cell and a co-op can act alike, a group of co-ops can form a larger co-op, so that cooperation among co-ops would be much like—the appropriate abstraction barrier would allow us to say *exactly* like—cooperation among cells.

This resolves, at least conceptually, the *scaling* question: It's all well and good to conduct a laboratory proof-of-principle experiment involving a few dozen agents, but how should several thousand agents be organized to manage a large distributed power grid? We respond that they should be organized as co-ops whose members are themselves co-ops.

This requirement implies new features. An agent representing a cell would provide an abstracted public picture of the cell. It would not be necessary to distinguish[9] a cell trying to join a co-op from a genuine co-op acting as if it were a cell trying to join a co-op. Fundamentally, cell capabilities and co-op capabilities need to be separated from one another and made available as separate packages. This means that a co-op could then take on the cell-level capabilities needed to be a member of a co-op, which would enable it to participate in co-op operations as a member element, i.e., as if it were a cell.

It also means that co-ops and cells could be members of the same co-op. Connections out of a co-op would be managed as if they were connections out of a cell, and co-ops could negotiate with one another and with cells in a larger glob or a larger co-op. Co-op policy would constrain the interaction.

Although a cell agent would have detailed information about cell contents and a co-op would generally be a much larger entity than a cell, this would all be hidden by the abstract interface. A large power system consisting of many thousands of microgrids would be organizationally fractal, in the sense that each of the pieces of the organization would be very similar to the organization itself; co-ops would be made up of smaller co-ops, etc., with the most primitive elements being single cells following co-op rules.

---

[8] This doesn't limit interaction among individual cells from different globs in any way.
[9] To be fair, an entitiy with whom one was interacting would likely identify itself as a cell or co-op, but to an implementor, the protocols, messages, and content information would be the same.

4.3. Organizating Principle Use-Case Application Notes

Two major questions about cell behavior are: "What happens within a cell?" and "What happens between cells?" To respond to these, we had elected to construct an agent coalition to manage the four-tent camp (Section 6).

We had to make a choice of how to implement the cells/globs/co-ops in the context of the four-tent simulation. There are essentially two possibilities: *One cell*: Represent the four-tent camp as one cell; *Four cells*: Represent the four-tent camp as four simple one-tent cells that function as a co-op. We selected the four-cell approach.

The primary argument in favor of the one-cell case is that we should first understand the operation of a single cell and only then tackle the more-complex issues of co-op management. The four-generator simulation certainly fit our definition of a single cell, and once we had a functioning cell, it would be fairly straightforward to replicate it and achieve a co-op. And we'd be more likely to have early success.

The counterargument was that it wouldn't be a very interesting success: Having constructed an operational four-tent cell would not particularly improve our understanding of how to operate a power system using agents, except in the degenerate case of a single cell and its single agent. We would answer only the first of the two important "cell questions. And, we reminded ourselves, cells were supposed to be simple, so the agent would barely be needed. This approach would also delay what we perceived to be the more interesting part of the project: The second "cell question" concerning between-cell behavior, involving multiple agents coordinating behavior among several cells.

In favor of the four-cell case, we argued that making four cells would require only re-executing whatever code we used to make the first one, so we should make it as simple as possible. We'd reach the stage where agents add value sooner. Since we had designed a simple scenario to start with, this was probably the simplest thing we could do that would begin to demonstrate the relevance of agents. The counterargument was that we'd be designing and building everything at once.

4.4. Agent and Organizing Principle FAQs

The cell/glob/co-op formulation is a valuable organizing principle for agents, because it anticipates the qualities put forward in the literature that make agents valuable in operating power grids. It also enables a response to several other concerns. In this section we outline these concepts as Frequently Asked Questions:

Q: Under what conditions should an agent be allowed to act autonomously?

A: An agent can act autonomously when it holds the belief that executing a particular action will either bring the system into a liveness condition or prevent the system from leaving a safety condition and when it is authorized by policy to execute the action.

Q: Why do agents act?

A: An agent of the sort described in this report acts either in direct response to an authorized operator or because its internal data is consistent with behavioral premises that so acting will increase the likelihood of satisfying a liveness condition (or reduce the likelihood of violating a safety condition) and is consistent with policy.

Q: How do agents interact with an infrastructure?

A: The infrastructures we are considering—primarily electric power, but also natural gas, petroleum, transportation, and so on—are *flow* infrastructures: they exist in order to get some commodity from many sources of the commodity to many usage points through a network of transmission links. Interaction with a flow infrastructure consists in a relatively small number of action types: Starting, stopping, or limiting the production rate of a source; and opening, closing, or constraining the transmissivity of a transmission link. Complex sets of these actions must be coordinated to maintain the "character and content" of the flowing commodity; for example, a stable A/C waveform is important for electric power and material content is important for petroleum. Coordination among the decisionmaking entities also enables efficient use of production and transmission resources.

Q: In what manner are the agents situated in the power system environment?

A: There are two ready answers: The more facile response is that the system design would incorporate sensors and effectors. These would include sensors for line current, generator output, and possibly load levels, and electronically controlled switches and breakers. Sources would need to be commandable as well. The second response, which is almost certainly more realistic, is that the agents would use data already present in existing SCADA systems, by which most of today's power systems are already managed. Since power systems are currently managed reasonably well, the notion of building a completely separate new agent-based system to serve a similar (if perhaps more deftly automated) function would be a hard sell unless significant, genuine advantage can be shown. Using existing SCADA elements would be challenging for implementers: SCADA systems are inherently centralized to allow a few human operators to manage a large area, but one of the primary advantages of an agent-based system is that the operation and management function is distributed among many agents at many locations. This conceptual collision is likely to cause engineering difficulties and compromises.

Q: Should there be an organization that represents, and whose job it is to handle, an islanded section?

A: Not necessarily. A glob that is cut up will devolve into smaller globs naturally, whereas a co-op whose network gets cut into islands should maintain oversight of all its islands in order that they continue to abide by co-op policy and in order that an organized re-grouping process can be followed at the appropriate time.

Q: Will there be a hierarchy in the agent system that's like the power grid hierarchy?

A: The co-op, and co-ops organized into co-ops with other cells, etc. forms a hierarchy that seems a reasonable way of organizing the existing power grid hierarchy given the penetration of microturbines and other microgrid technology. Power and voltage are high when being distributed over long distances and drop as power is distributed from big generating sources down through different levels of distribution substations to end users, so the existing grid is a tree, at least in terms of levels. The combining or gathering of distribution lines into transmission lines is also suggestive of a cell abstraction barrier and placement of co-op boundaries.

— This page intentionally left blank —

5. Microgrid/DER Deployment Considerations

There are several possible approaches to deploying a microgrid, and these vary in two primary ways: the presence or absence of an existing conventional power infrastructure to tie into, and a control strategy that's either centralized or distributed. We discuss each of the four possible configurations, and note some of their advantages and disadvantages, some of the operational and ownership issues involved, and how they change as they grow. This is followed by an overview of industrial and residential deployment considerations.

5.1. Centrally controlled; connected to the main grid

This is generally what is referred to by the term "DER"; it's distributed from the point of view of primary power production. On the other hand, when a set of microsources is run with a central controller and tied into the main power infrastructure, it's like a big power plant in every way but size. Although outwardly comparable to a microgrid, this kind of installation is not operated in a distributed fashion and is, conceptually speaking, only a little more complex than to operate than a single generator.

Control is centralized either in the sense of having all sensor and control channels connected into a common controller, or in the sense of having the generators all reporting to and obeying a single controller over a network. There are technical issues with ensuring that the generation of power is coordinated—having suitably configured droop rates, deciding when to turn generators on or off to follow load, establishing a policy to determine the generation in response to local load, etc.—but the phase and frequency of the system can be made to follow the bulk power supply, and the detail that the power is being generated by an array of microturbines or other such devices can be largely abstracted away since there is little difference in the transmission paths between the individual generators and the load.

This satisfies the general desire of the utility operator that a power resource be simple to operate. A study by the Consortium for Electric Reliability Technology Solutions (CERTS) [Kueck] states: "the [electric power] utility does not want to be burdened with additional control issues … The first goal is for the microgrid system to appear to the utility only as a controllable load. Local electric utilities are justifiably concerned about uncontrolled voltage regulation."

Such micro-plants can be used to ensure continuity of power when access to bulk power becomes constrained, or serve as local supply for peak-shaving, load reduction, or power conditioning (in which case access to bulk power is used as the backup rather than vice versa). Typically they are run as fixed, homogenous installations; often they are engineered to meet specific requirements associated with a known load's characteristics, and new generation capacity is either not added, or added to match expansion of the load. Because the plant and load are most likely owned by the same organization, expansion and reduction of the facility probably does not have to be coordinated between very many parties.

5.2. Centrally controlled; not connected to the main grid

When a centralized microgrid has its connection to bulk power severed, or if it is constructed with no connection to a bulk power network, it must operate in such a way that the load is met within predefined tolerances (or selectively shed), and the generators must maintain phase and frequency in synch with one another and within the tolerances of the load to handle. Because there is a central authority, control of the official phase and frequency can be centrally managed. For homogenous plants the difficulties associated with centrally controlled power systems have been addressed, but unplanned addition or removal of capacity remains difficult. This is referred to as "plug and play" (often shortened to "p&p"), and [Lasseter04] has shown that, if certain conventions are followed, a microgrid can be operated as a collection of plug-and-play sources (loads are essentially plug and play by nature; proper operation of a power system is mostly about managing sources in the face of undisciplined load variation).

In small to mid-scale cases, the valid "island" when bulk power is severed will be determined in advance because of the nature of a centrally owned or managed facility, the coupled nature of a centrally controlled plant, the load it is designed to support, and the requirements of how the load must be supported in this condition. Islanding portions of the national power grid remains a far more difficult problem, not only because of technical challenges of coordinating the significant alterations in load and generation as the network undergoes substantial topological changes, but because of the legislative, procedural, and financial issues that come into play for a collaboratively managed system that is run at every moment under the auspices of pre-arranged contracts.

5.3. Distributed control; connected to the main grid

This is a collection of microsources that are independent of one another, not collocated, operating autonomously, and connected independently to the bulk power network. Because of the inertial capacity of the environment, phase and frequency are established by the bulk power network. The overt question is how much power each of the generators must produce under the assumption that excess or deficits will be accommodated by the bulk power network. In general these systems are not centrally owned, which means generation and compensation have to be negotiated. When the bulk power system is the national infrastructure (few producers; many consumers), this may be effectively brokered by the bulk power network, but for systems that are distributed with the intent of supplying a certain set of distributed loads, and for systems that must maintain certain conditions (for safeguards, etc.), collective internal negotiation will be a greater issue. This case, and the corresponding disconnected case, will be of interest to facilities that want the benefits of a locally maintained microgrid but want the system to handle changing topological configurations (the connection and disconnection of portions of the facility from one another, for example) gracefully and do not want the liability of a central point of failure in the power supply.

5.4. Distributed control; not connected to the main grid

This is the most difficult case. Phase and frequency have to be controlled collaboratively, since there is no main grid, and power generation has to be negotiated without the benefit of a large inertial bus to absorb or supply the necessary excess. These systems could be flat or hierarchically organized, with local power subsystems ganging together to form semiautonomous agencies that interact with one another at an amortized level, leaving the agents within to negotiate how the details will be accomplished in support of the higher-level contracts made with other organizations. The multiagent system discussed in this report was designed to accommodate this case.

5.5. Industrial Park Microgrid Design

The four-square-mile industrial park in Figure 2 is suitable for businesses that employ technologically advanced manufacturing processes such as paper mills and semiconductor fabs. Microsources will be distributed throughout the park such that they are located at the lots of the largest consumers, those with the greatest sensitivity, or those that can benefit from cogeneration. A fault-tolerant underground distribution network would provide power to each lot from the collective set of microsources. The following power system questions arise (IEEE Std 141-1993 and 493-1997 shed some light on these):

- Would an underground distribution network to be fully installed in the early phases of the park development, or would it be extended as occupation contracts are signed?
- How is the distribution network typically sized if it's installed in advance?
- Are critical & non-critical consumer nominal load profiles available to size the microgrid?
- What components are necessary in the distribution network?
  - 3-phase power cables
  - Transformers
    - Voltage conversion for cross-park transmission
    - Harmonic suppression
    - Down-line voltage regulation
    - Phase compensation
  - Switchgear, Breakers (automatic protection and controllable)
  - Surge Arresters, Lightening protection
  - Static Var Compensators
  - Power factor correction
  - SCADA: Voltage, phase, power flow monitors and controllers, communication cables.
- Mathematical thermo-magnetic models that predict when an automatic breaker will trip given a power flow profile are needed. Typical curves appear in IEEE Std. 141-1993, p. 197
- What is standard industry practice in distribution network topologies to:
  - Provide component and line fault tolerance[10]?
  - Avoid circulating power?
  - Support non-critical load shedding before critical load shedding?
  - Satisfy park occupation changes over a period of years?
- What human activities and interfaces are needed to maintain the operation of the microgrid on a 24x7, rain-or-shine basis?
- What are the more common failure modes in a distribution network and relative probabilities of occurrence available? For example, "More than 90% of voltage disturbances in utility lines are single-phase voltage sags caused by momentary line-to-ground faults in distribution systems." [Venkataramanan] Types of failure modes are discussed in IEEE Std. 1159-1995 and IEEE Std. 519-1992. Component reliabilities are covered in IEEE Std. 493-1997.

[10] Electric service capital costs factors are: distribution 50%, transmission 30%, and generation 20%, so at what point in distribution would redundancy occur?

Figure 2: Notional Industrial Park Microgrid Layout

This microgrid would require a large number of microsource. The paper mill (70 MW) or the semiconductor fab (65 MW) alone would require thousands of microsources. The whole park would need seven thousand 60-kW microsources, plus reserve units, to supply an estimated power demand of 420 MW. Average density would be about one microsource per half-acre.

The installation cost for gas-fired microturbines is approximately \$1000/kW and O&M costs are around \$1k/microturbine/year. Fuel cost would be on top of this. These units are 25% efficient if only the power output is used. Thermal plumbing can add considerable expense, but a commercial heat exchanger adding as little as \$7k to the installation cost can boost overall efficiency to 75%. Given bulk power @ \$0.09/kWh and natural gas @ \$0.44/therm (both expected to increase as this is being written), the payback period is about 5 years for electric power only and 2½ years if the thermal output is utilized (cost data from [ENVIR]).

Consider a neighborhood of 3000 homes in an area of 1.5-1.75 square miles. This neighborhood would require a maximum continuous load capacity of approximately 58 MW (= 240 volts * 100 amps * 80% continuous * 3000 homes) and peak capacity of 72 MW. Approximate average load can be estimated at 20% of maximum continuous load capacity or 12 MW. This neighborhood would need 400 30kW generators at average load and 2400 generators at peak. A reasonable estimate is 4 homes per generator, or 750 generators. Given the high cost of distribution, at what level in the distribution hierarchy would redundant circuits be provided? The IEEE Std. 141-1993 manual provides some information about this.

6. The "Four-Tent" Use Case

For this project we defined a "military encampment" setting for use in describing how the agents should react to various situations. The microgrid consists of four generators, each paired with a critical load and a non-critical load. Each load has its own load profile (i.e., operates independently of the other loads). A communal bus connects the four subsidiary power systems. This enables power from any generator to be used at any load. This network is shown in pictorial form in Figure 3 and as a PSAT power-flow diagram form in Figure 4. Power-flow values and voltages in Figure 4 are in per-unit (p.u.) terms with a system base of 1 MVA. The generator buses are operated at 480 volts. Each of the four generators is a microsource comprising a parallel combination of a generator and battery storage on a DC bus behind an inverter controlled by a digital signal processor. The microsource is treated as an ideal power source with a finite settling time unspecified at this time. Because battery response time is very short, we suspect that if this system were ever realized the inverter control loop would be the limiting factor.

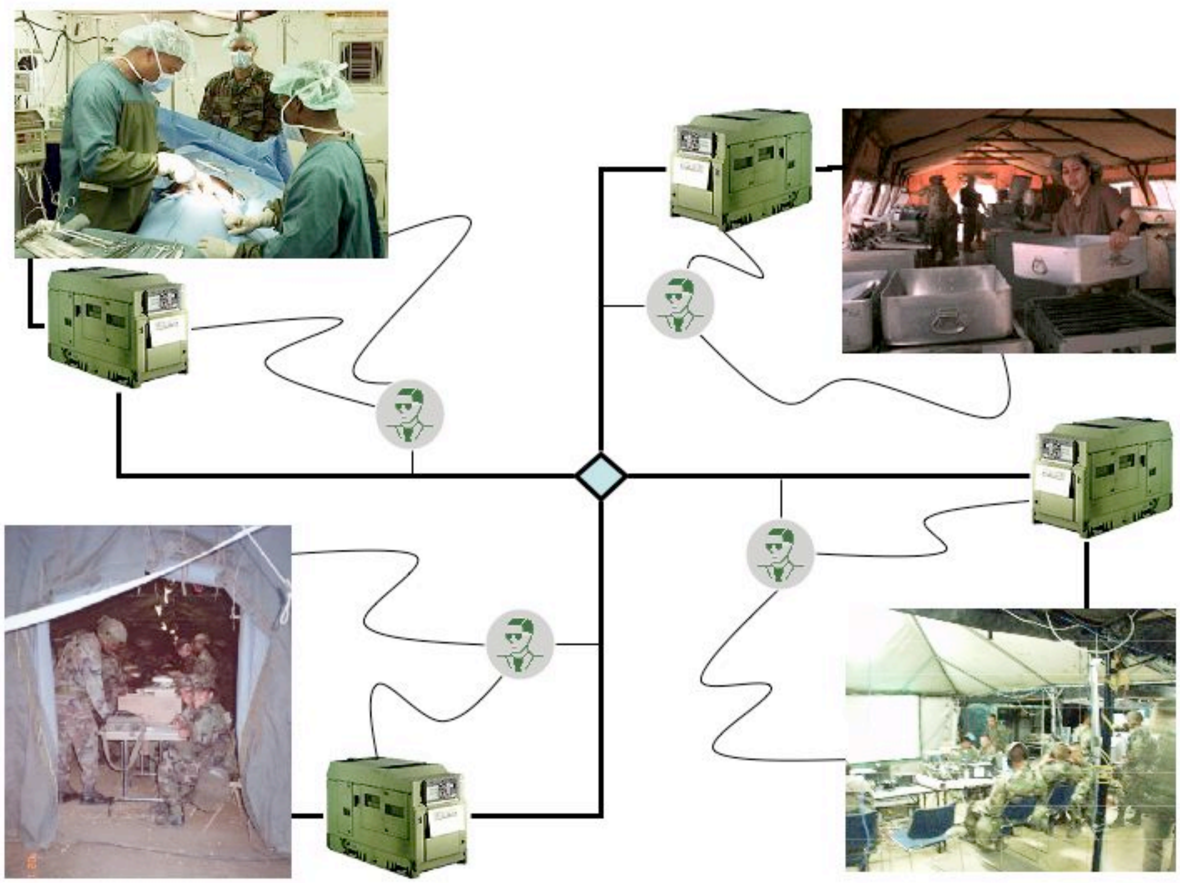


Figure 3: The Four-Tent Scenario (clockwise from upper left: Hospital, Mess, Headquarters/Communication, Barracks)

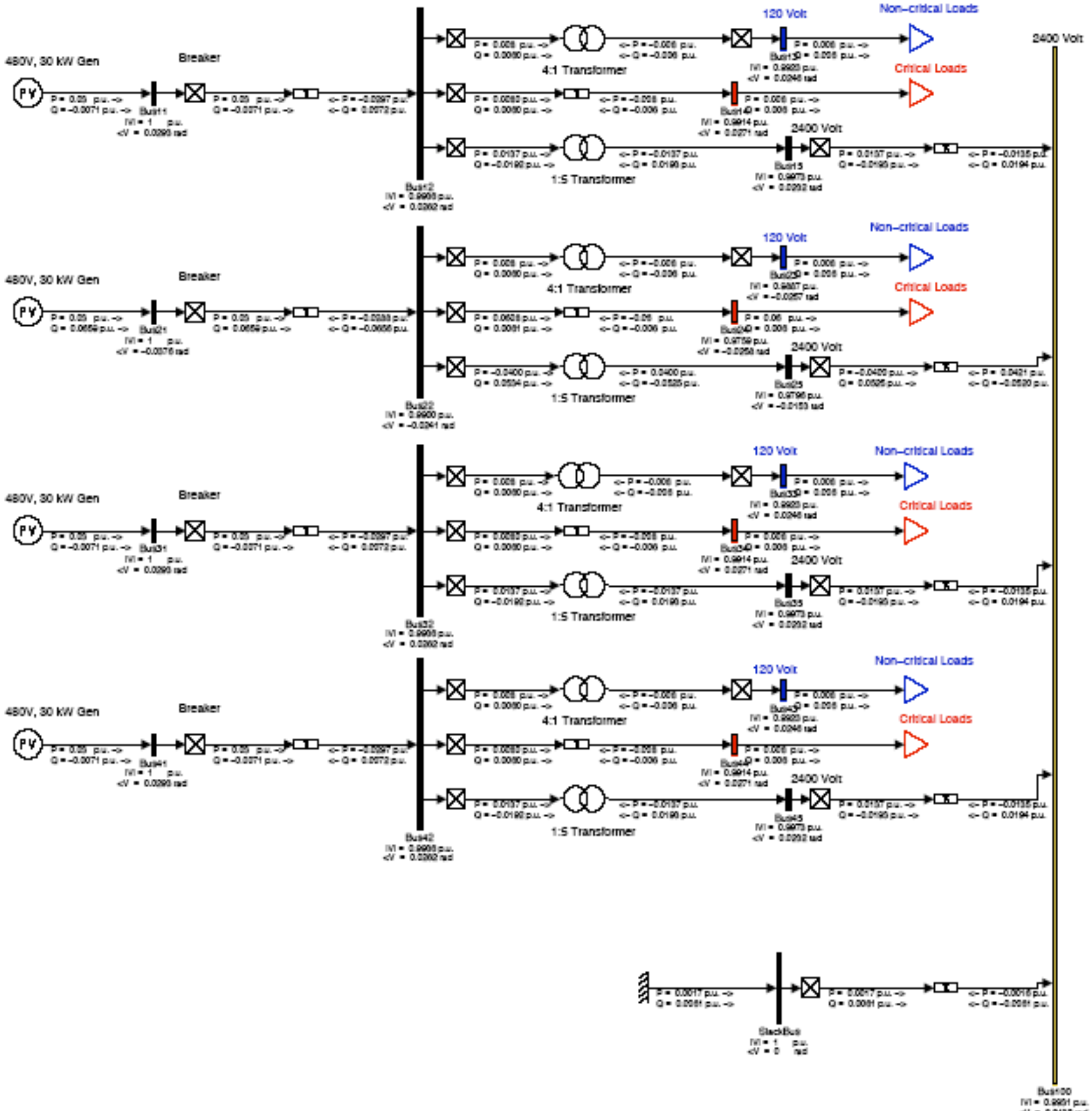


Figure 4: **Four-Tent Microgrid Schematic**

A cell can be a relatively complex entity, so a single agent could manage the four-tent system as a cell. Conversely, each tent could be considered a cell, and the four “tent agents” would operate as four cells and interact as a co-op. Based on the reasoning outlined in Section 4.3, we chose the latter approach. Each generator and its associated loads and distribution network represents one of the tents: headquarters/communication, hospital, mess, and barracks. Since each tent has its own managing agent, generator, loads, and distribution network, it can operate independently of the others. Because all four generators are connected to a common bus, the four power subsystems can operate cooperatively.

6.1. Demonstration Objectives

The primary goal of the project demonstration is to exhibit agents managing and operating a simulated electric power microgrid in such a way that observers could perceive the benefits of using agents to do this. Strategies expected to be effective included providing contingency plans, producing ideal solutions, or solutions better than the state of the art, and improving the general power network. An ancillary goal is to demonstrate the improved security that's possible when Sandia's agent technology is used.

6.2. Demonstration use case descriptions

These descriptions outline the agents' operations when starting up, configuring, and operating the system and when responding to selected external events.

In the use case selected for demonstration, the system removes a generator from the network as requested by an authorized human. Its description in this section is the most detailed. A similar second use case that we did not demonstrate is to detect and respond properly to the sudden unplanned absence of one of the generators. The descriptions discuss details such as the protocols agents execute, the particular agents and Virtual Domain Authorities (VDAs) executing the protocols, when the agents run which decision algorithm, how they come to consensus on the results, when they randomly elect a winner of several options, etc.

The authors recommend that as development of a microgrid-operating agent system proceeds, the developers construct a repository of use cases that span the state space of the operational system: its failure modes, operations, permissible actions, and before/after states for events of interest. Generator addition, transmission line failure, load spike, and dropped load are some of the use cases that need to be considered in addition to the two above.

6.2.1. System Start-up

This is a complicated element of any distributed system that is fundamentally necessary and, just as fundamentally, not very interesting as a demonstration. Bringing the agent coalition to active status involves powering up hardware, loading and activating software, activating the network, establishing agent intercommunication links; and handshaking and group formation. Once active, the agent system must activate and determine the status of its sensors and effectors, establish connection with the electric power system, determine the power system's status, and, finally, assume responsibility for the power system based on authorization to do so and the appropriate status of the agent system, the sensor/effector system, and the power system. All the above must be accomplished in the correct order and under only appropriate conditions.

6.2.2. Generator Start-up

A user turns on a generator and the device is detected by the protection system and cell operator. The presence of this device is reported using the plug and play (P&P) protocol and the Strategic Planner VDA initiates planning the use of the new resource. The individual agents run the same decision algorithm independently to choose new set points across the network. The VDA enacts consensus and signs the plan. This plan is distributed using the appropriate protocol and the tactical controllers begin to act on the plan at the appointed time. Status information is set through the monitoring system and presented in the GUI.

6.2.3. Improving Service

When there is surplus capacity in the system, unsatisfied loads should be considered for service. Selected loads should be tested for current state and a plan for new generator set points created, distributed, and enacted.

6.2.4. Generator Shutdown

Generator shutdown is the use case we developed for demonstration. A user goes through the request protocol in order to take a generator down for maintenance. The strategic planner is a Virtual Domain Authority (VDA) with respect to all four tents. The individual agents plan for the new set points of the remaining generators given the active generation policy. Since policy may prohibit certain sustained levels, the agents need to know the duration of the period before the generator can be returned to service. In any case the agents may arrive at a situation where generation is at maximum allowable levels and cannot satisfy existing load, and so the discrepancy must be noted, consensus on it reached, and a list of possible load-shedding solutions compiled (equal-priority loads should be selected by random election so decisions about what to shed are fair). Once this is done, generator set points are again computed, and this may iterate if the system is still not operable with that shed load due to transmission limitations, etc., until a suitable plan is found. This plan is then signed by the VDA and the plan distribution protocol is enacted to ensure everyone knows about it. The plan is put into motion at a prescribed time, which needs to be part of the plan. Start time is selected based on when maintenance needs to start and how long the shutdown will take, plus some margin for error, and at the appointed time the tactical agents will do their parts. The new state of the shut-down generator and generation/distribution/use across the network should be reflected in the GUI, and the agents should report plan execution results through the monitoring system.

6.2.5. Generator Failure

When a generator fails, the protection system and cell operator detect this and shed non-critical load. The *raise alarm* protocol is run to make the entire system aware of the problem and contingency plans, if any, are implemented. Assuming there is no local backup generator that could be used, the strategic planner will come up with new set points for the remaining generators and may need to identify loads to be shed[11]. It may be appropriate to have crisis policies in place to enable faster planning and more conservative results. At the tactical level, some of this may already be done with a local non-critical load within a cell being shed to mitigate the failure. Once a solution is developed it is distributed and enacted.

[11] As with a planned shutdown, the remaining generators might be over-subscribed.

7. Modeling the Four-tent Microgrid

This section describes the construction of the Four-tent microgrid Matlab model, using the SimPowerSystems (SPS) toolbox and Simulink graphical simulation environment. We were assisted in this by [Saadat], which helped us to understand the behavior of the simulated devices we were observing and to make reasonable choices from among the many possibilities that presented themselves, and [Ong], which provided significant insight into the behavior of the Matlab environment.

7.1. Modeling considerations for the Four-tent Microgrid

A microgrid can be modeled in either software or hardware. Simulation of detailed software models is CPU-intensive. Simulating a single ideal 60 Hz voltage source and resistive load using Matlab's Simulink program requires 1/2 second of wall clock time per second of simulated time on a 600MHz G4. Simulating a DC-AC inverter model using IGBTs (Insulated Gate Bipolar Transistors) and behavioral control models can take several hundred seconds of wall clock time per second of simulated time. An ideal microsource model with an external battery store is being constructed based on published details from NREL's 4-year microgrid program. Hopefully, the performance and representation of this model will be sufficient to support initial ABCDIR experiments. It's also possible that the ideal model can be used to support the development of experiments and then follow this up with a higher fidelity simulation that could run over days or weeks. We mention in passing that, whatever approach is selected, the simulator code must be altered to allow agents to interact with the simulation.

An option for high-fidelity simulation is to use massively parallel computation and a circuit simulation program like SPICE (Simulation Program with Integrated Circuits Emphasis), a general-purpose analog circuit simulator.

Yet another option is to construct a hardware model of the microgrid. COTS Uninterruptible Power Supplies (UPSs) would be used as microsources. This would entail reverse engineering to gain access to the inverter control inputs and decoupling the frequency from the AC input. The COTS UPS would limit us to single-phase modeling. In their hardware simulator, NREL purchased COTS inverters and is driving them with programmable DC power supplies. In either case, a substantial data-acquisition system is needed to enable the agents to interact with the hardware model. Other components, such as transformers and loads, are also required. This approach is costly and consumes a fair amount of lab space.

7.2. General Microgrid Operations for the Four-tent Case

The microgrid should be configured with a single load-following generator and any number of fixed output units. Note that two or more identical load-following generators cannot be interconnected since their close-loop controllers will beat against each other. The load-following generator should have its local breaker closed first and allowed to settle with its local load before adding additional loads. We are told that Caterpillar specifies this settling time to be 10 seconds.

The load-following generator is a single point of failure in this grid configuration. Without the load-following generator, there is no means to adapt to the resulting generation/load imbalance. The options for such a situation include:

- Pre-elect one generator with sufficient capacity to switch over to load-following mode automatically in the event of load-follower failure;
- All tents island (i.e., disconnect from one another; which would leave the tent connected to the failed load-following generator without power. This suggests the lowest priority loads be the ones directly connected to the load-following source.), convert individually to load-following mode to service their individual tents, communicate to reassign the load-following role, and reconnect;
- Halt power production, reassign the load-follower role, and restart.

At best, recovery time is limited by the transient generator response and the load deficit, although some grid architectures could be more tolerant than others. If the recovery time is on the order of several seconds, it is possible that an agent collective could detect the failure and issue corrective commands in real time. With shorter recovery times, policy-directed, closed-loop controllers would be more appropriate. Commercial firms with experience in this area (e.g., Capstone, Caterpillar) are likely to have developed strategies to handle this sort of failure in their ganging controllers, such as the Capstone Power Server.

From a modeling and simulation perspective, one can't simply connect a load to a fixed-output generator that has zero mechanical input power because the generator will attempt to match the load by altering its frequency and voltage. To start a real-life fixed-point generator, one would ensure that the load-following generator has sufficient capacity to carry the fixed-point generator cell's local load and then close the breaker connecting the fixed-point generator's cell to the microgrid. After an appropriate settling time, the operator would set the power limit of the fixed-point generator and close its local breaker, allowing it to take load according to its set level. As an alternate procedure, the operator would set the fixed-point generator's power level consistent with its cell load and close the local breaker between the fixed-point generator and its cell. After the cell has stabilized, the operator would close the breaker connecting the cell to the rest of the microgrid. Either way, a deployed system should have a synchronization controller to oversee the final connection between generator and the microgrid.

7.3. The Four-tent Model High-Level Requirements

The microgrid developed for the project supports an isolated military camp composed of four tents: Headquarters/Communications ("HQ"), Barracks, Mess, and Hospital. Such a camp might occupy six to twenty-five acres. Distribution grid power lines will be relatively short, so we have chosen to ignore capacitive effects and limit distribution voltages to 2400 volts. The camp model is shown in Figure 5.

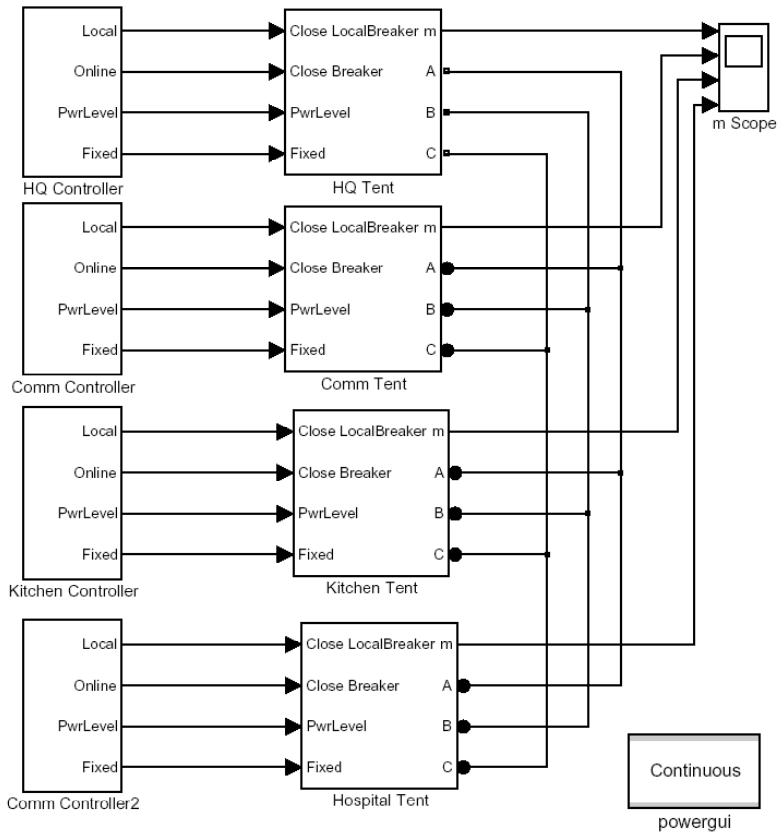


Figure 5: Four-Tent Military Camp Model

7.4. Four-tent Model Description

Each tent has a local load, primarily real, but it can also be specified to have reactive elements (households and small businesses rarely have significant reactive loads). Each tent also has a local generator, which can be configured to have fixed output power or load-following behavior. Figure 6 shows the fixed-output generator tent model.

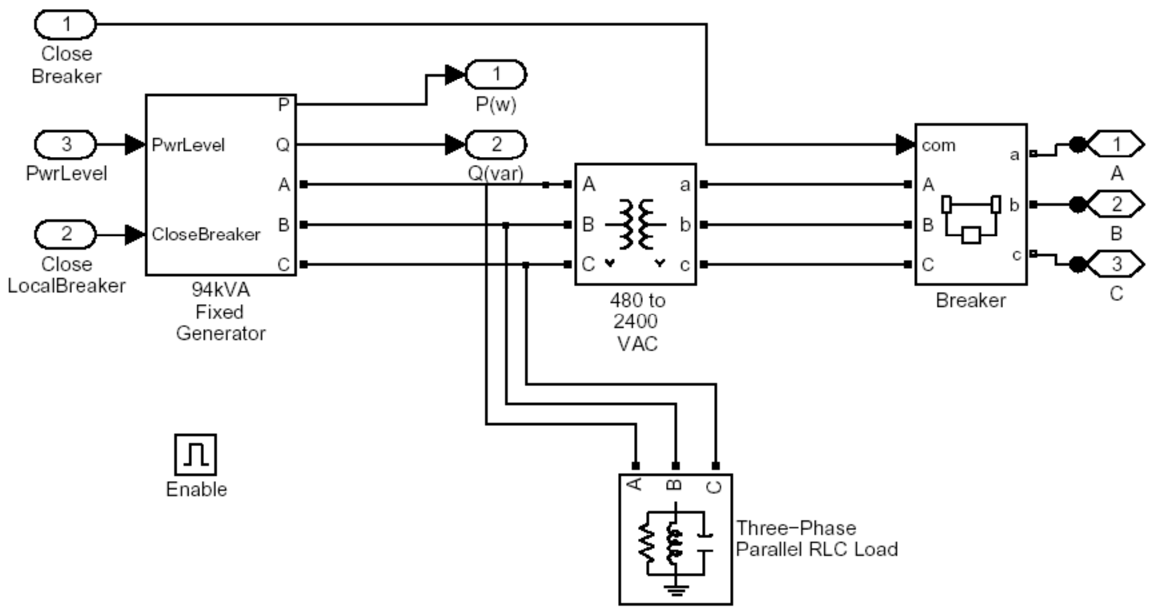


Figure 6: **Tent with Fixed Output Generator Conditional Model**

Two generator models were incorporated in the project microgrid model. The first is a fixed-output unit developed at Sandia. It is based on Caterpillar diesel generator parameters, but without reference to the Caterpillar controller design. The model uses Matlab's *SPS Synchronous Machine* primitive (2 poles, salient configuration), an *SPS Excitation System* (regulates reactive power and voltage output, also called an Automatic Voltage Regulator, AVR), and a custom Load Frequency Controller (LFC) to regulate real power and frequency. The LFC takes a power level input of zero to 93.8 kVA via a rate limiter that limits the full-scale slew to about 11 seconds. A Woodward EGCP controller controls the diesel generator. The controller has provisions to control the slew-rate, and the 11-second rate maintains an over-damped response, which apparently worked best with the Caterpillar. The model outputs consist of the three 480 V RMS phases and real/reactive power measurements in pu. A three-phase breaker was added to the model and the power measurements were converted to watts and vars. This model appears in Figure 7.

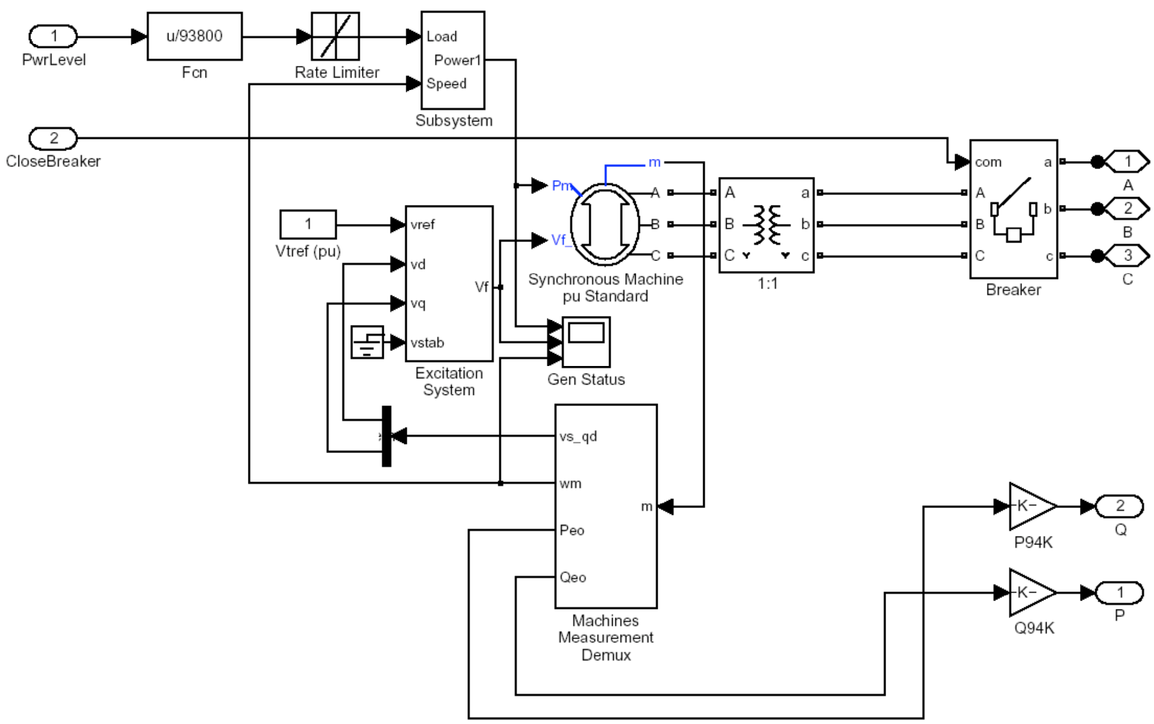


Figure 7: **Fixed Output Generator Model**

The second generator model is a load-following generator from an *SPS* example file. Its synchronous machine parameters were replaced with those from Shawn's fixed-output model. This model includes a combined prime-mover, AVR, and LFC subsystem. That subsystem consists of a Governor & Diesel Engine subsystem (LFC) and the same *SPS Excitation System* AVR. The LFC provides closed loop control of the synchronous machines mechanical power input while the Excitation system provides closed loop control of the field voltage. The load-following generator model is shown in Figure 8.

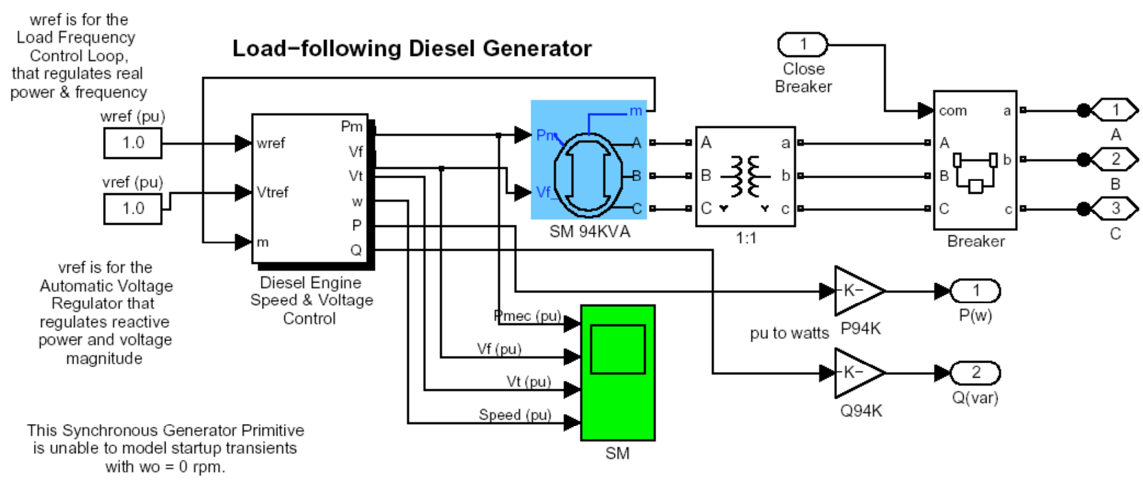


Figure 8: Load-following Generator Model

Each generator was instantiated in its respective fixed-output / load-following Tent model, consisting of the generator, a local three phase parallel RLC load, a 480-2400 V transformer, and a microgrid breaker. Control inputs were provided to isolate the generator from its local load and transformer and from the microgrid. The fixed-output generator tent has an additional power level input. Each tent model has the property of being conditionally executed in support of a generic tent model, as shown in Figure 9.

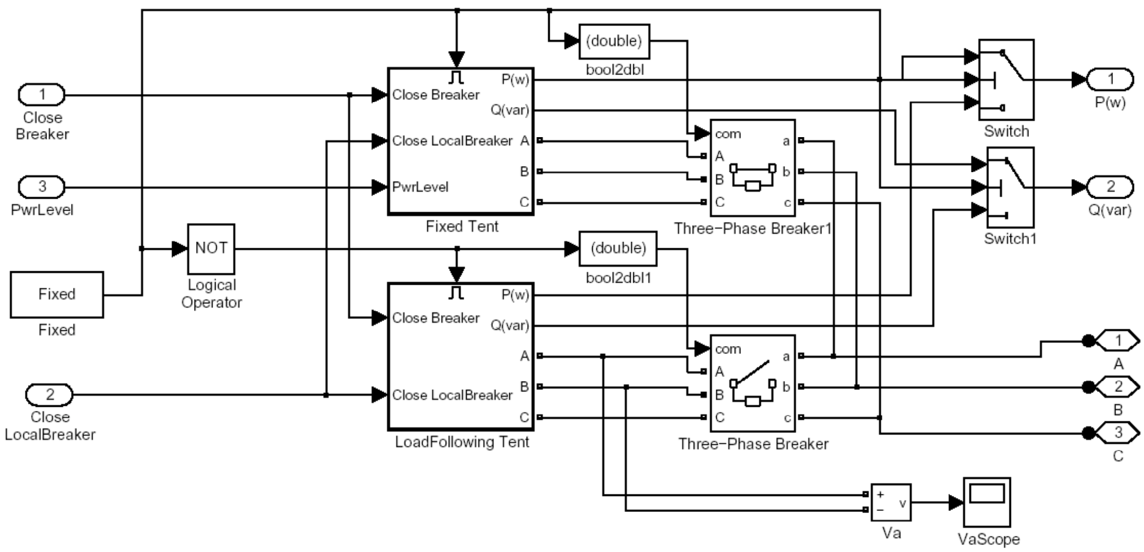


Figure 9: Generic Tent Model

The generic tent model was added to the hierarchy to support convenient switching between the fixed-power generator and the load-following generator simply by checking a checkbox on the tent parameter form. Ultimately, a controller will want to specify this sort of operation. Wiring four tents together using ideal buses created a first iteration of the microgrid shown in Figure 5. Each tent is driven by a controller subsystem that consists of a graphical signal editor and waveform viewer.

Note that this model makes no effort to synchronize the closing of the generator breaker. A suitable controller would accept the “close” command and then ensure that the voltage and phase of the generator match that of the bus within a specified tolerance before actually closing the breaker. This enhancement should be considered for follow-on work.

7.4.1. Scenario Implementation Notes

Unfortunately, the generic tent model would not compile. Apparently, various *SPS* primitives use *Simulink GoTo/From* primitives that provide interconnects not specified by the user. We think Matlab uses a global node namespace that that allows variable name collisions between the tent models. We altered the generic/fixed/following design into a hybrid generator model that can switch between fixed and follow mode. That model appears in Figure 10, while Figure 11 shows the resulting generic tent.

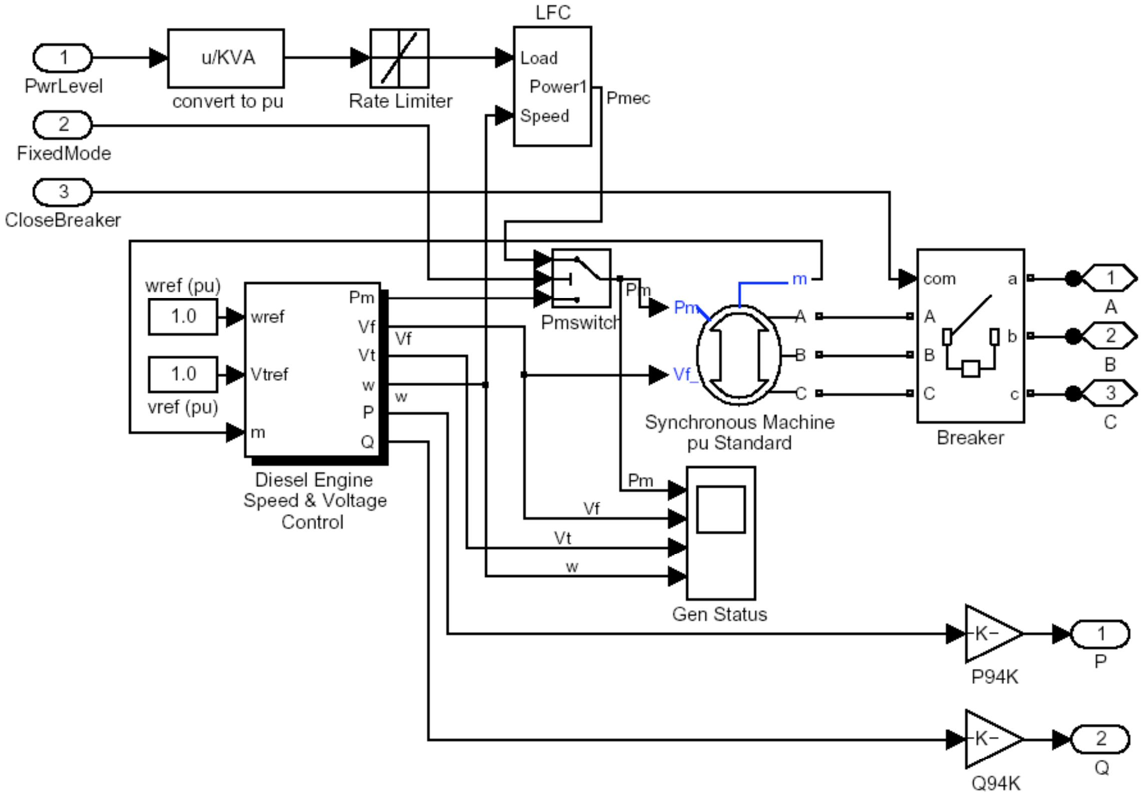


Figure 10: Hybrid Fixed/Load Following Generator Model

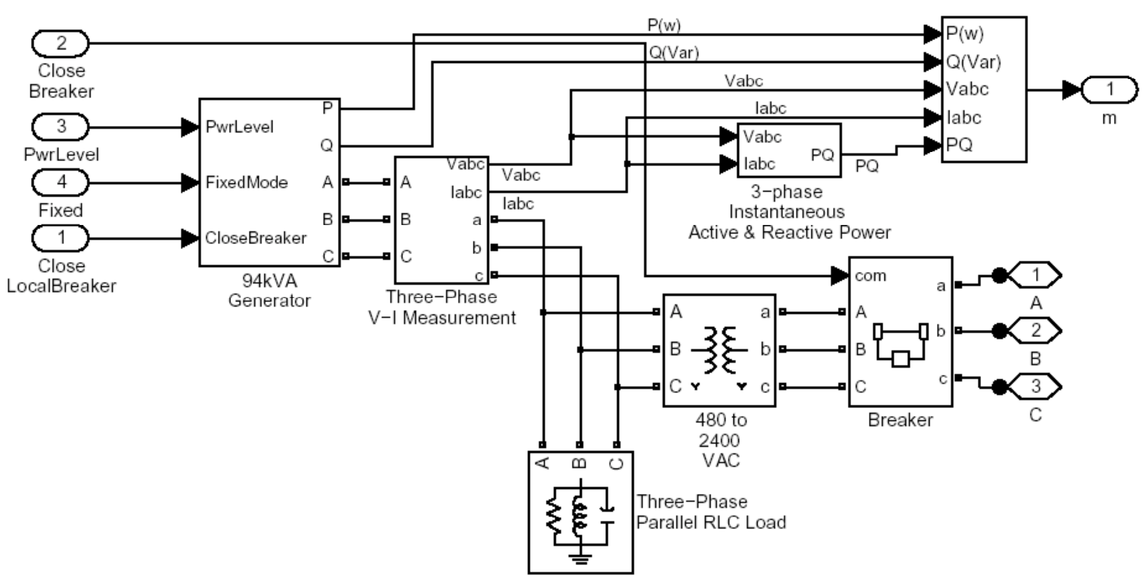


Figure 11: Generic Tent with Hybrid Fixed/Load Following Generator Model

The models were placed into a *Simulink* library as shown in Figure 12. However, the new hybrid models still exhibited the compilation failure. Replacing all library references with instantiated copies enabled the model to compile. This engenders a hierarchical design of unique block and primitive instances. This makes model construction cumbersome and error prone; edits made to a given component must be manually made to all instances.

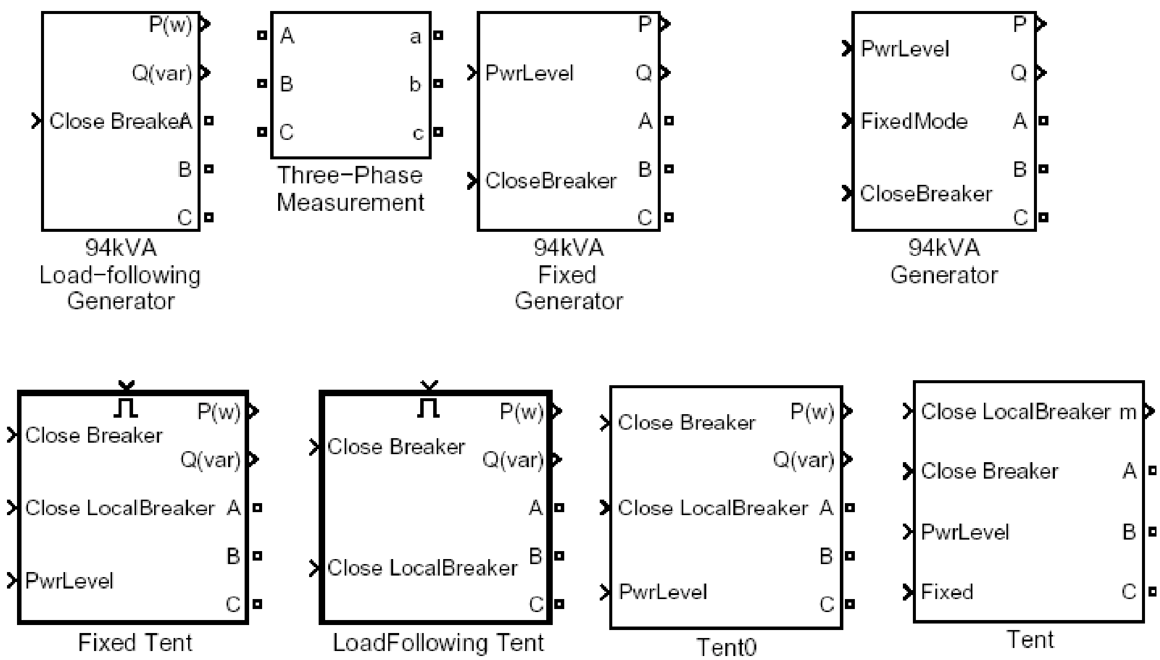


Figure 12: Microgrid Library

The 3.x version of *SPS* introduces a power-flow interconnect type distinct from the classic *Simulink* interconnects. Model icons represent the power-flow interconnect terminals with a square terminal and normal *Simulink* control and measurement interconnect terminals with an angle-bracket symbol. Control/measurement arcs have a solid arrowhead on the destination model icon.

In *SPS* version 2.x, all interconnects are classic *Simulink*. Sandia's Caterpillar generator model was created with version 2.x and it had problems executing in 3.x until it was translated to a 3.x model.

7.5. Continuous Simulation of Four-Tent Camp

This continuous simulation was run on a dual-1.3-Mhz-processor Apple Xserve. It took 12,332 seconds to simulate 90 seconds of circuit operation, about a 137:1 ratio. We implemented a phasor simulation to achieve better performance. The phasor simulations took 2,070 seconds, a ratio of 23:1. Figure 13 shows how the circuit was changed to measure the power flow between the tent and microgrid. Discretization to improve further improve performance was considered, but we did not pursue this.

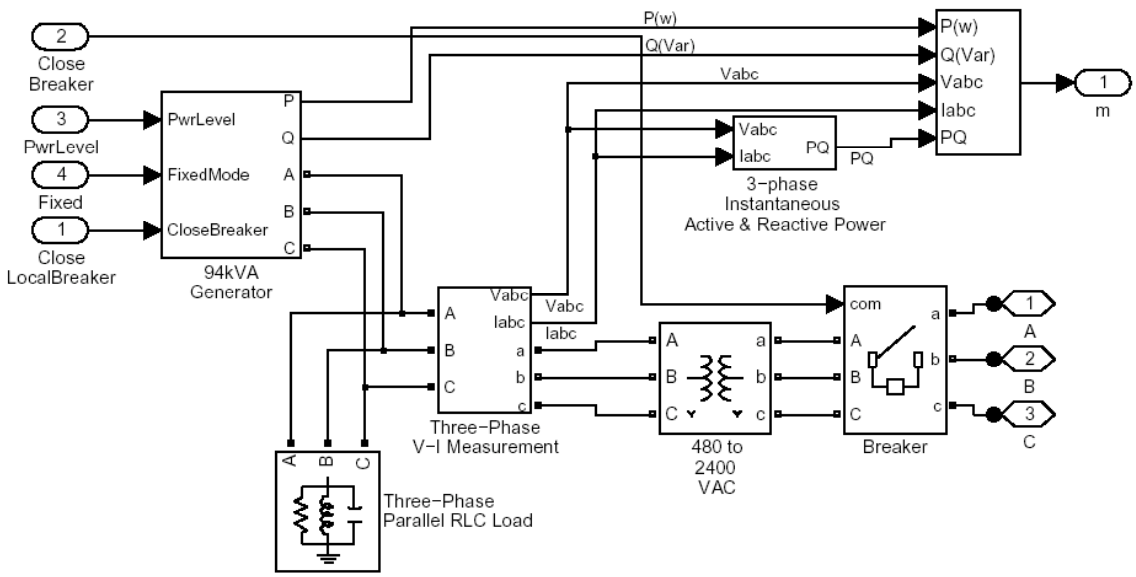


Figure 13: Generic Tent Model with Powerflow Measurement

Figure 14 shows how the measurement bus is routed to a scope measurement subsystem. Also the control stimulus was placed into a single signal generator to make it easier to visualize the timing during editing. Note the "follow" indicator on the HQ tent[12].

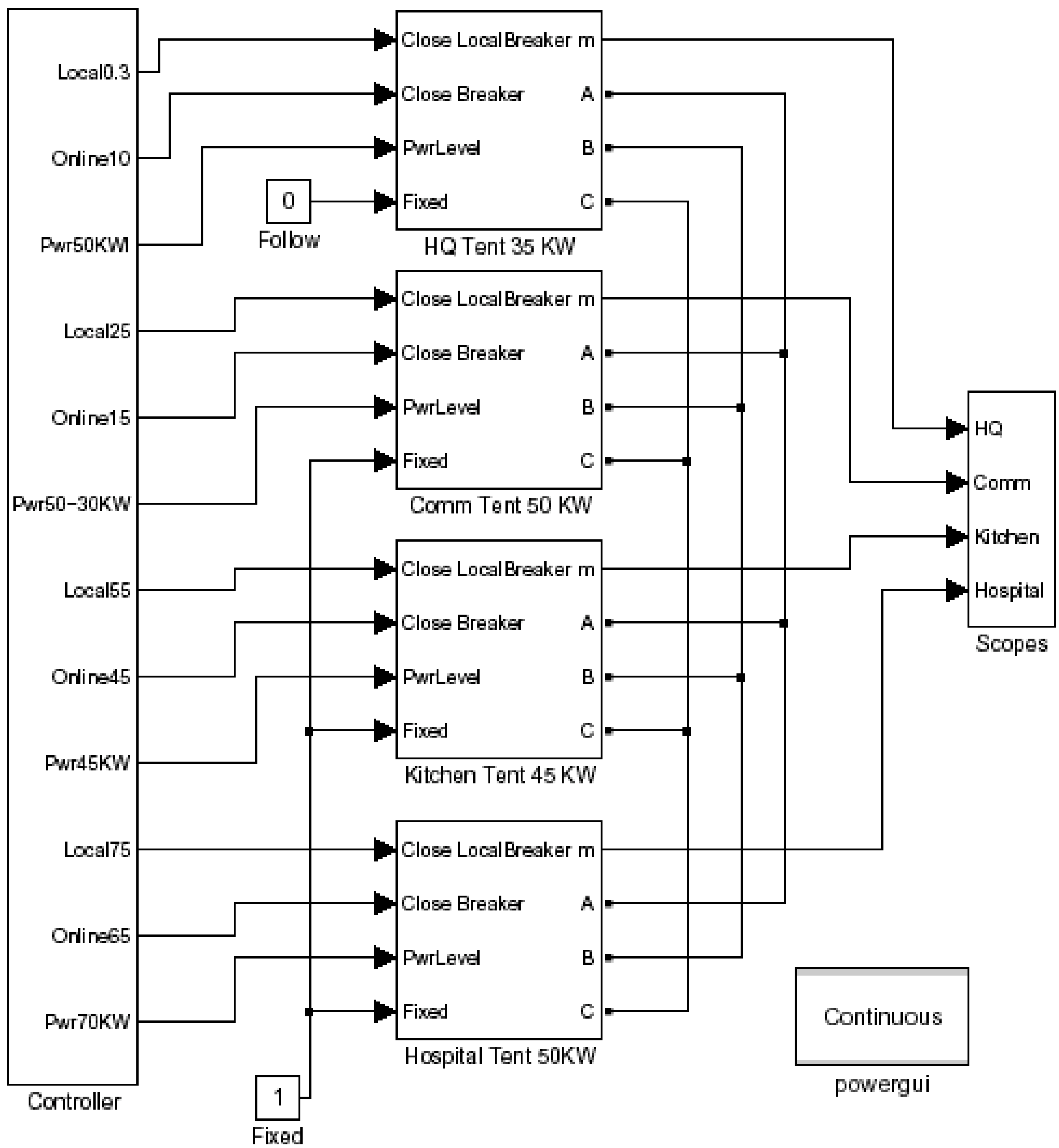


Figure 14: Four-tent Camp with Power flow Measurements & Combined Flow

[12] The "HQ" block in the figure is the Headquarters/Communication tent; "Comm" is the Barracks; "Kitchen" is the Mess.

Figure 15 shows the control stimulus for the simulation. The HQ tent was configured to have a load-following generator and the others configured with fixed generators. There are three waveforms per tent: local breaker control, online (microgrid) breaker control, and power level. The first three are for the HQ tent, the next three are for the Barracks, the next three for the Mess, and the final three for the Hospital. The order is identical to that of the Controller labels in Figure 14. The HQ generator connects to its load at 0.3 sec and connects to the microgrid at 10 seconds. Since it's in load-following mode, its power level is ignored. Then at 15 seconds the Barracks tent goes online (connecting its local load to the HQ generator). At 25 seconds, the Barracks generator sets its power to match its local load and goes online. At 35 seconds the Barracks generator drops its power by 20 kW, forcing the HQ generator to increase its output by that much. At 45 seconds the Mess's 45 kW load is connected. This should swamp the HQ. Its generator goes online at 55 seconds to supply the 45 kW. At 65 seconds, the Hospital 50 kW load attaches and its generator goes online at 75 seconds with a 70 kW output, reducing the load on the HQ generator.

Figure 16 shows the power flow at the four tents, HQ through the Hospital. Blue is real and red is reactive power. At 15 seconds the Barracks Tent online breaker closes placing its local 50 kW load across the HQ generator. Note that these PQ measurements show only the power flows between tents and the microgrid, thus HQ's local 35 kW load is not shown.

The 5 kW, 60 Hz oscillations are primarily due to resonance between the generator and the transformer's magnetization reactance. Figure 17 shows the effect on the three-phase currents. We suspect these are modeling artifacts.

Returning to Figure 16: At 25 seconds, the Barracks generator power is set and its breaker closes. Notice the 10-second slew rate. At 35 seconds, the Barracks generator reduces its output by 20 kW and the HQ generator picks up the deficit. At 45 seconds, the Mess connects its 45 kW load to the microgrid. The HQ tent seems to be providing this power just fine even though this exceeds the generator capacity by 16.2 kW. At 55 seconds the Mess generator kicks in and the HQ loading falls back in range. At 65 seconds the Hospital load is added while its generator goes online at 75 seconds, relieving the load on HQ by 20 kW.

Figure 18 shows the three-phase voltage at the 25-second event where the Barracks generator goes online. The voltages seem to synchronize and stabilize quickly.

Figure 19 shows the HQ generator status in p.u. From the top down: mechanical power, field voltage, generator terminal voltage, and rotor velocity. In the beginning, the HQ generator is supplying its 35 kW load or about 35/94=0.37 pu. This agrees with the HQ's mechanical power. At 15 seconds, the Barracks's 50 kW load is added or 85/94=0.90 pu. Again the mechanical power agrees. At 25 seconds, the Barracks generator begins supplying 50 kW, and HQ's mechanical power stabilizes at 0.37 pu. The overloads are seen when the Mess and Hospital loads are applied. Terminal voltages remain at 1 pu except when the breakers change state.

Figure 20 shows the generator control for the Barracks tent. Its power input stabilizes to .53 pu around 35 seconds, agreeing with HQ's mechanical power.

Figure 21 and Figure 22 show the Barracks's LFC and waveforms.

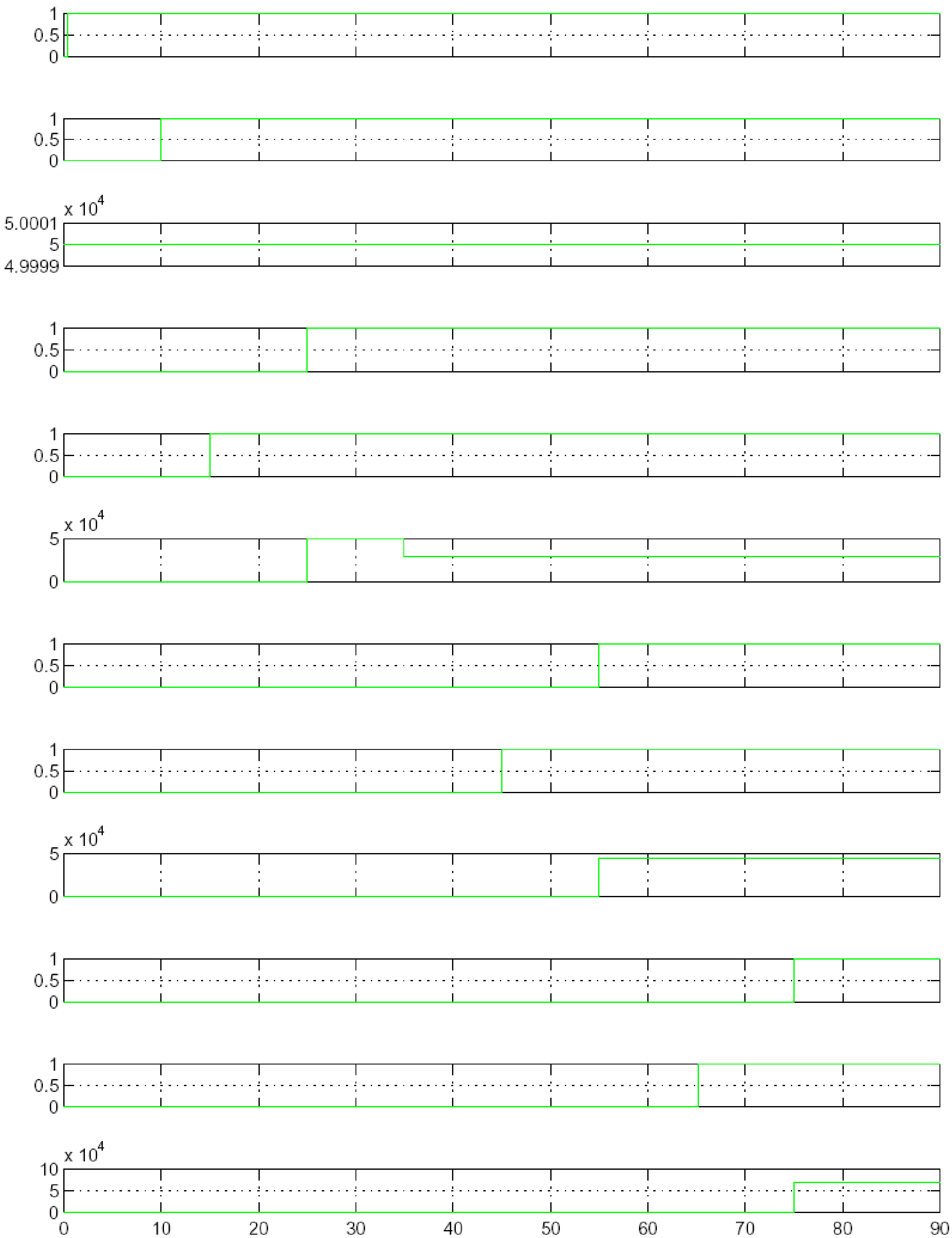


Figure 15: Four-tent Control Signals

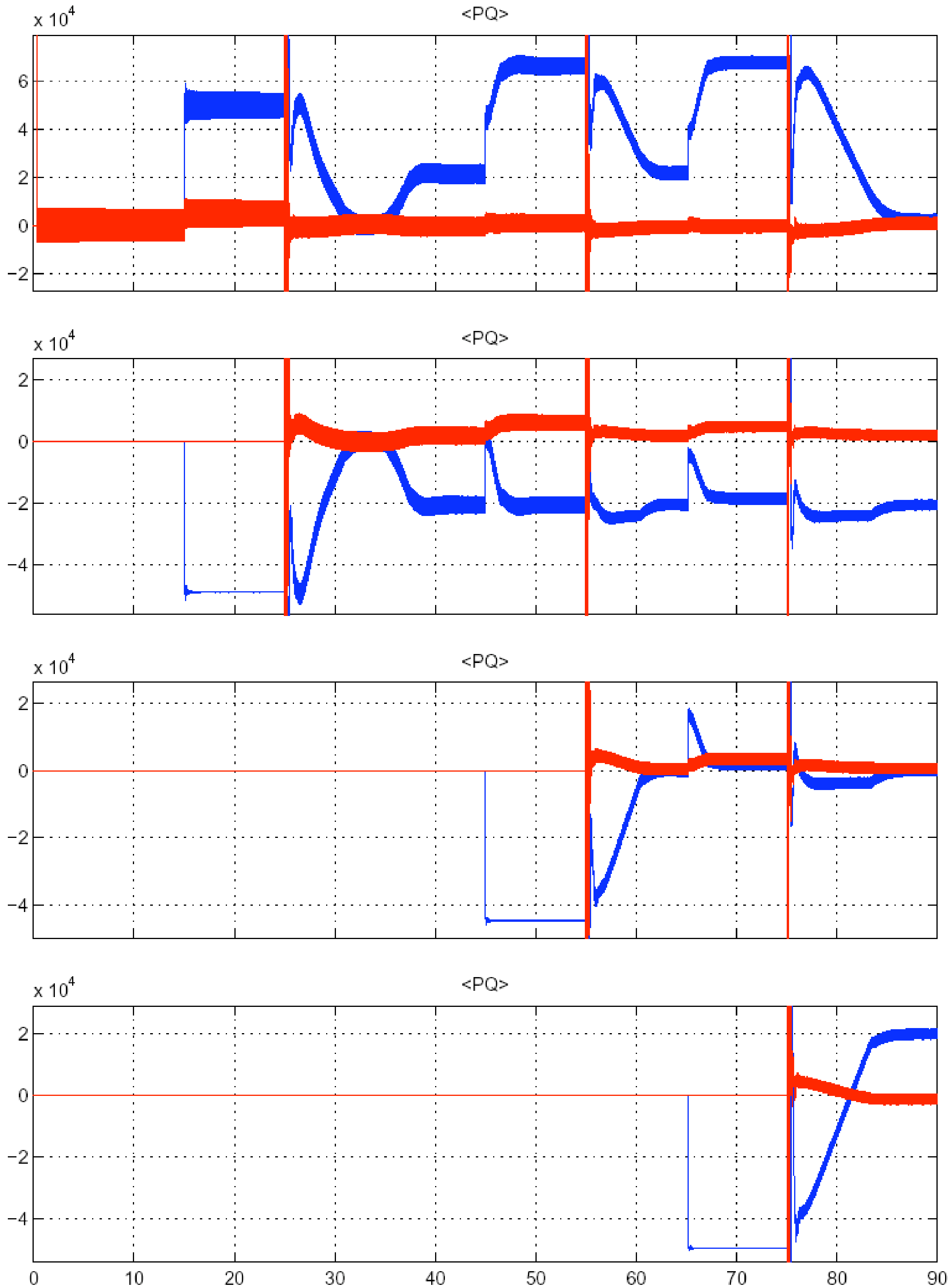


Figure 16: Power flow at the HQ, Barracks, Mess, & Hospital Tents

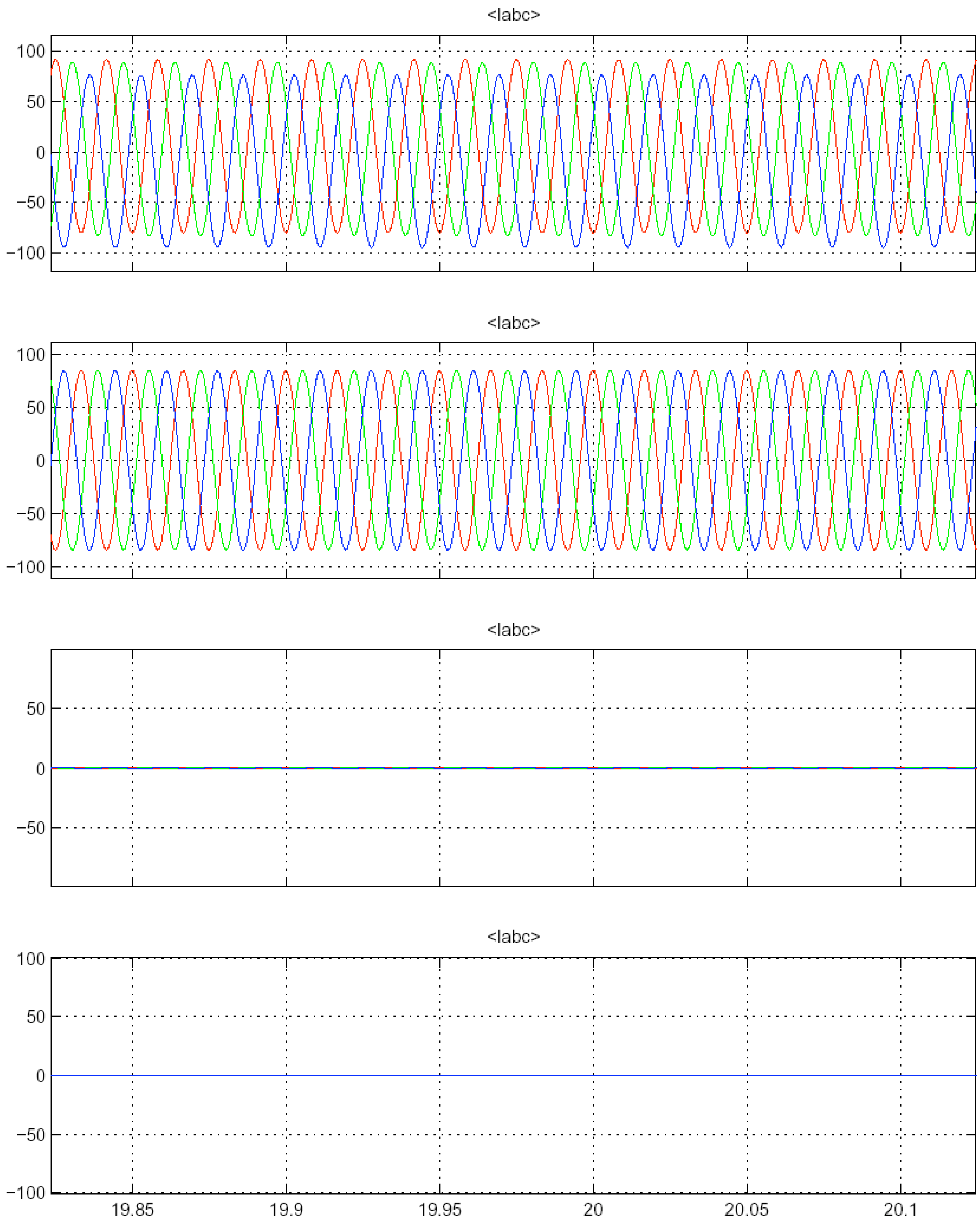


Figure 17: Three-phase current showing unbalanced currents

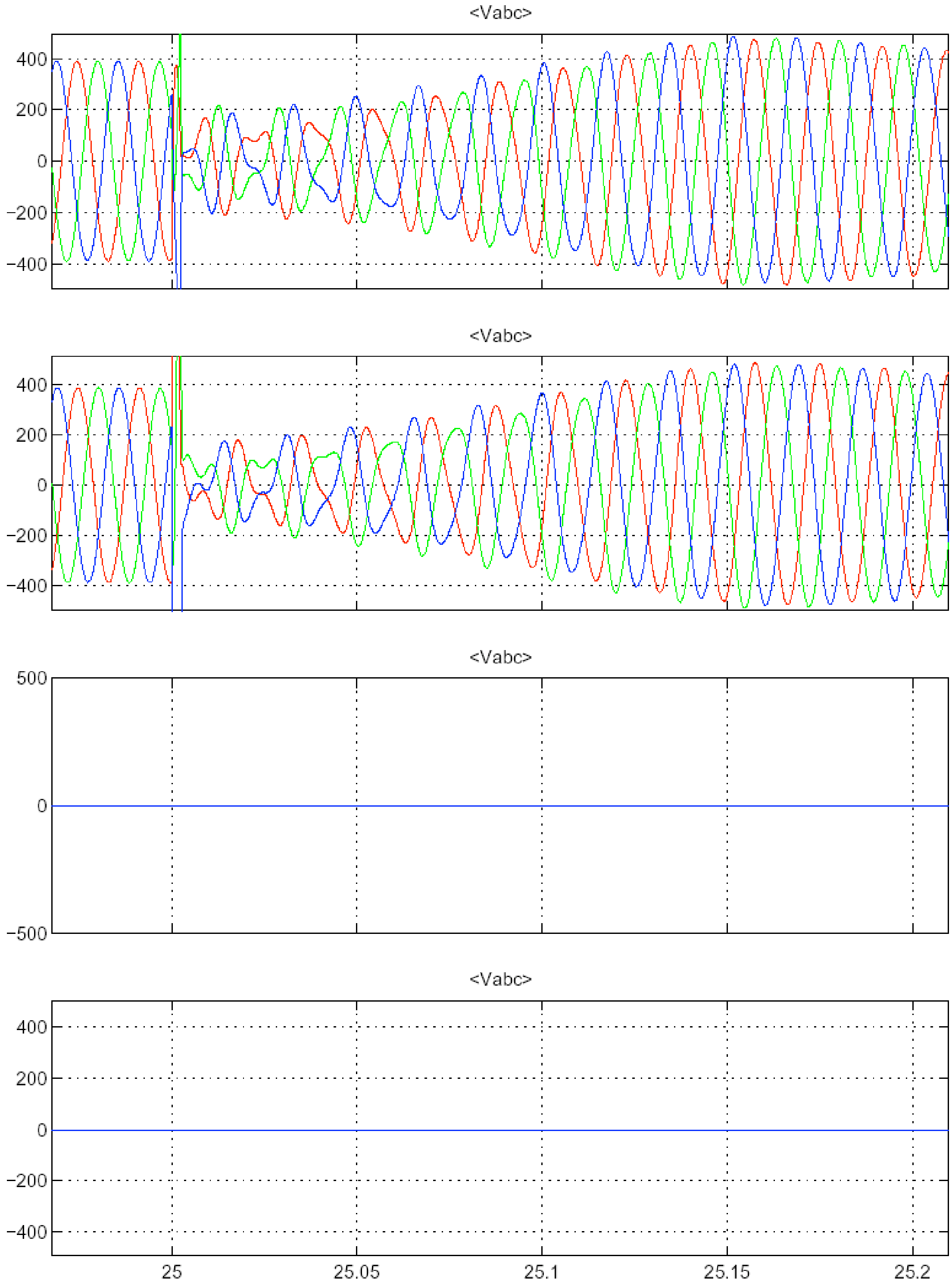


Figure 18: Three-phase voltages at local load near the 25-second event

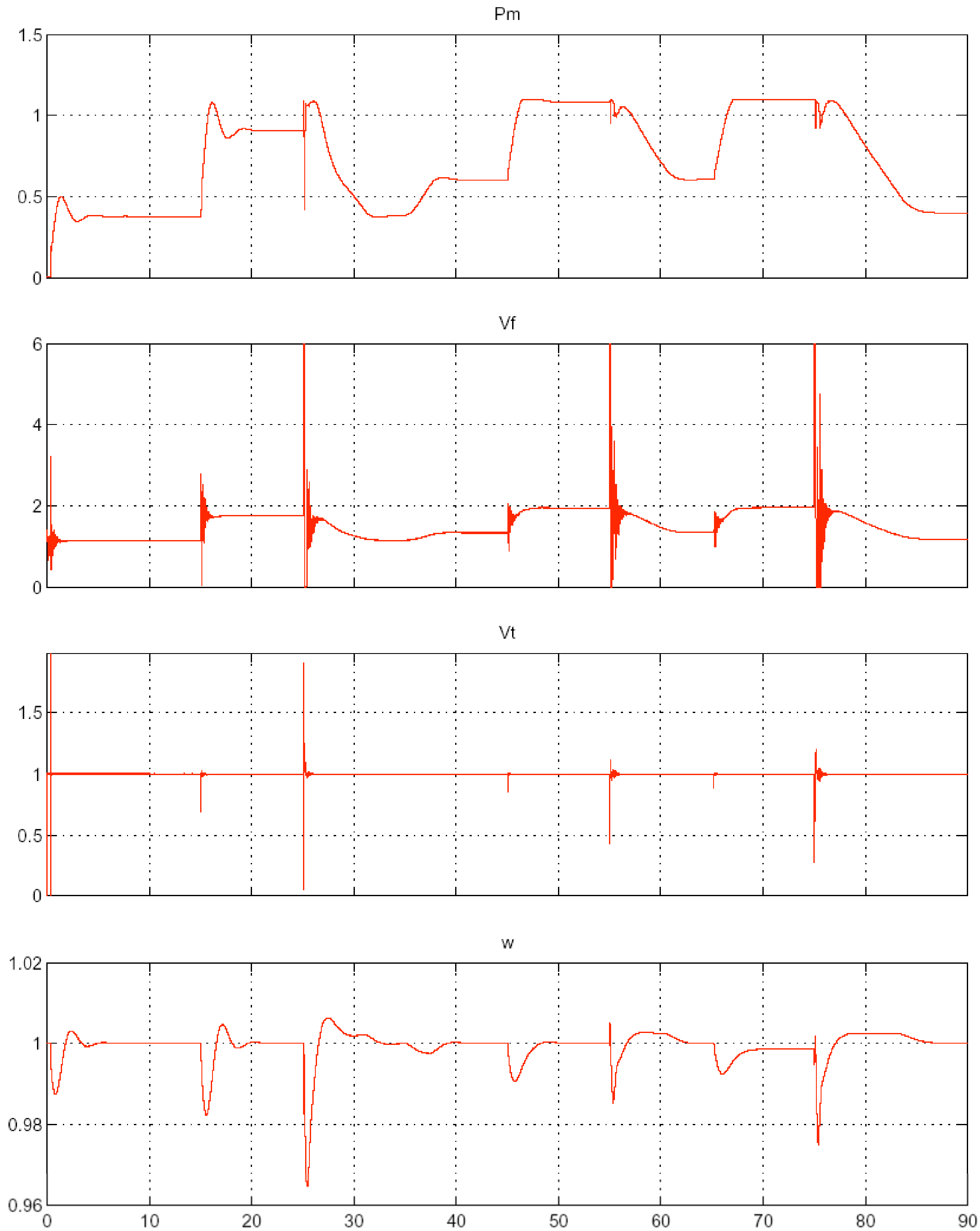


Figure 19: HQ Mechanical Power, Field Voltage, Terminal Voltage, & Rotor Velocity

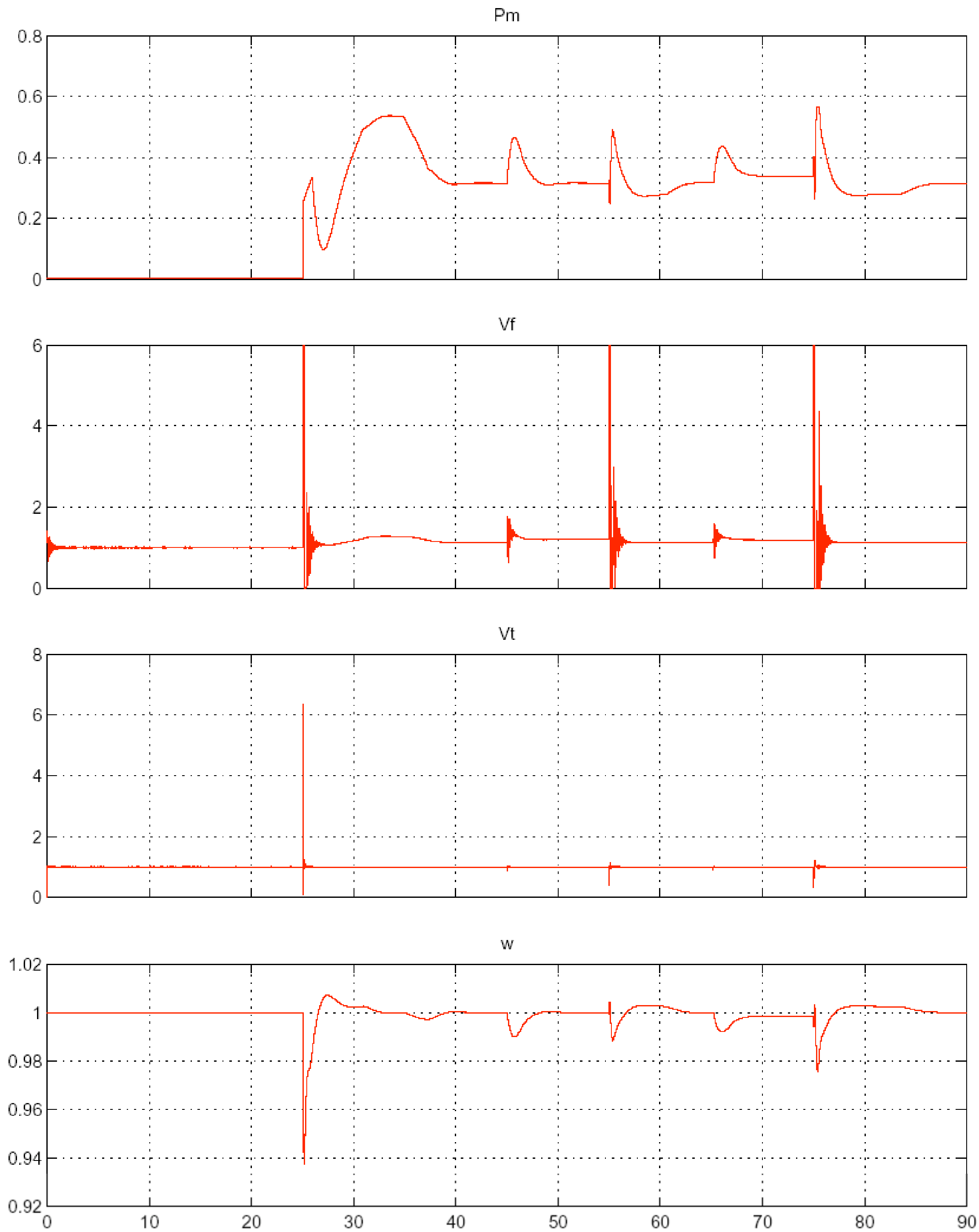


Figure 20: Barracks Mechanical Power, Field and Terminal Voltage, & Rotor Velocity

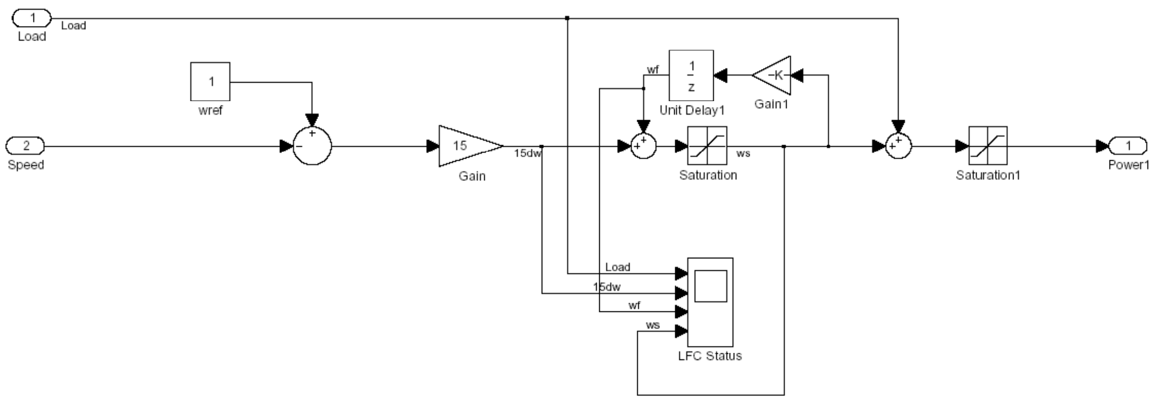


Figure 21: Barracks LFC Model

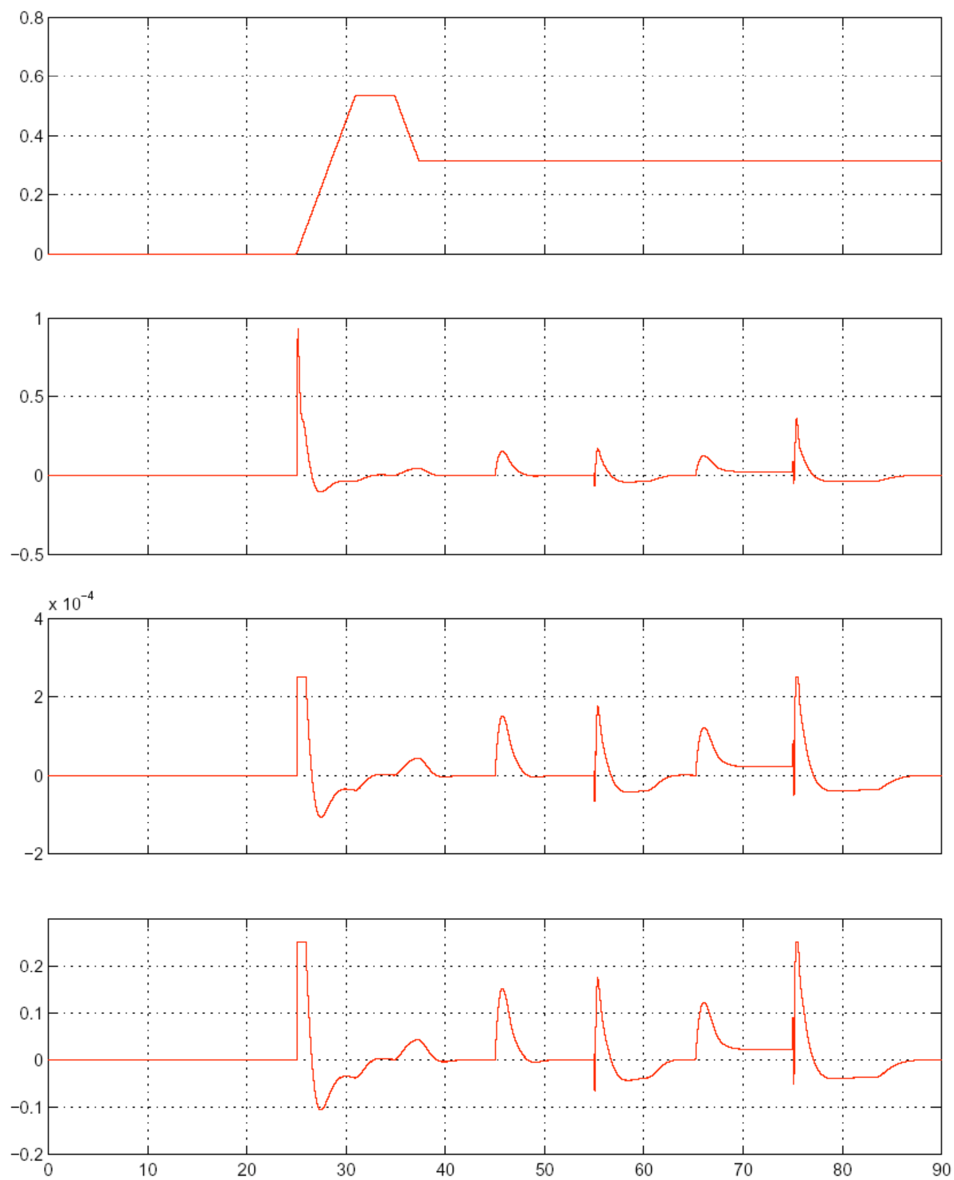


Figure 22: LFC Waveforms

7.5.1. SPS Synchronous Machine Primitive Issues

A modeling difficulty is related to the synchronous machine stator phases being modeled with current sources. Current sources cannot directly drive purely inductive series loads. This situation is remedied by adding parallel resistance to the circuit. In the Caterpillar generator model this was remedied by the inclusion of a delta-wye 1:1 transformer. In the four-tent model the transformer was removed because the snubbers in the local circuit breakers perform that function.

The *SPS Synchronous Machine* model, following [Kraus], has inputs of mechanical power (the product of mechanical torque and rotor angular velocity) and field voltage. The model outputs a large number of measurements including angular velocity. With zero field voltage, zero mechanical power, non-zero stator load, a 0.5 second inertial constant, and -0.6 friction, the rotor at synchronous speed slows down, passes through zero, and continues in the opposite direction. A continuous-time 9-second simulation took roughly 5 minutes of wall-clock time. A phasor simulation is much quicker but gives vastly different results: rotor velocity changes from 1 to +0.977, versus crossing zero as it did in the continuous simulation. It appears that this primitive can't be used to represent large deviations in rotor velocity from synchronous speed, e.g., starting from 0 rpm.

7.5.2. Simulink Implementation Notes

The Scope viewer autoscaling output requires one to periodically press the autoscale button during the run, which will chop the view of a monotonically changing waveform. After the simulation completes, one can zoom in or jump to full view, but scrolling and waveform measurement are not provided. The zoom-out command is found in the context popup. Library blocks will not compile if they include an instantiated Scope. There is no mechanism to import the workspace of a previous run and view its waveforms using the Scope viewer. Also, if a scope is omitted, the simulation has to be rerun after it is added.

The Matlab graphics in this document were printed as color postscript, but sent to a file instead of a printer. The postscript was edited to replace the line color of yellow lines, which are hard to see on white. For example,

```
/c8 { 1.000000 1.000000 0.000000 sr } bdef
```

sets the color to yellow; changing this to:

```
/c8 { 0.000000 0.000000 1.000000 sr } bdef
```

makes the color blue (RGB model). We used *macps2pdf* to generate PDF files from the postscript. The file is then opened in *Adobe Acrobat Reader*, cropped to the appropriate rectangular selection, and copied and pasted into the document.

Unusual Simulink windowing behavior (program freezing, clicking on window sends it to back, unresponsive window) occurs fairly frequently in *Gnome*, less often under *OroborOSX*, and rarely under *Panther* (Mac OSX 10.2).

A Matlab simulation is timed by entering the commands `tic; sim('modelname'); toc`, however, doing so may limit one's ability to interact with scopes and models in comparison to launching the simulation from the menu bar.

8. Power Management Agent System Design

In designing the power management agent framework we followed the Gaia methodology as extended by [Zambonelli] to open multi-agent systems. This approach focuses on the questions that need to be answered to construct an agent-based system, in particular on defining the organizations that need to exist. The segments of the approach are ordered such that the context for each stage has been provided in large part by previous stages.

8.1. ABCDIR Environment

The first step in designing an agent system is establishing the environment in which the agents will operate. In a physical system like the electric power domain, sensors and effectors are the standard means by which agents interact with the environment and such a mechanism will be transparent and assumed throughout the design.

In general, agents sense, affect, and consume pieces of the environment. We have noted when environment elements have parameters that are needed to reason about the system's behavior but may not be available.

Electrical Sources

Reads: voltage, current, frequency, real & reactive power, RMS values, temperature,

Changes: max power output level (% of rated capacity), connect/disconnect, on/off

Electrical Loads

Reads: real & reactive power, frequency, voltage, (static impedance?)

Changes: connected/disconnected

Lines – Distribution lines will not typically be observable by the system, but the system needs information about them to reason about them. For example, they sag when they get hot, so the system needs to know their temperature. This can be estimated from air temperature (from weather data), power flow through the line (from a power flow estimation model), and the line diameter and material.

Parameters: resistance (short-line model), maximum rating

Switches

Reads: real & reactive power, frequency, voltage, open/closed state

Changes: Open/closed state

Transformers

Parameters: power rating, impedance, step ratio

Sensors

Reads: sensor state-of-health

Thermal Sources

Reads: instantaneous thermal output. This could be computed by applying an efficiency factor to the input energy flow into the heat-making device.

Thermal Loads – Although power loads can be sensed by, e.g., frequency droop, the system cannot sense thermal loads and would need to be told to serve a thermal load at a given time and place. The system might track behavior and learn to predict thermal loads.

Parameter: thermal impedance

Reads: thermostat setting, temperature

8.2. ABCDIR Agent Roles

An agent *role* is one aspect of what an agent is supposed to do in an organization. The agents in this system are required to assume many roles at one time or another, and individual agents may assume multiple roles. These roles are associated with specific pieces of functionality that enable the agent to provide a particular service for the group. The agent roles that we identified are: System Point of Contact (POC), Architectural Advisor, Strategic Planner, Strategic Monitor, Tactical Monitor and Controller, and Device Protector. Each role has associated functionality. Human interaction with the agents is abstracted in a distinguished generic role called a *User*. This role is not played by an agent, but is used to describe and define the necessary external interactions.

8.2.1. User

Users add and remove devices, propose configuration changes and scenarios, and view information about actual or predicted system state based on real or hypothetical states. Users can also enable and disable individual sources, based on the premise that a user can have information for which the system has no sensors, e.g., that an inactive generator is ready to be returned to service.

8.2.2. System Point of Contact

An agent in the System Point of Contact (POC) role presents the current system state to a user through a UI, receives requests from the users to perform certain actions (such as setting maintenance schedules, enabling and disabling devices, etc.), and passes user commands into the system.

8.2.3. Architectural Advisor

An agent in the Architectural Advisor role deliberates among alternative planning policies; and potential sites to add sources, loads, lines, and sensors to the system. In this capacity the Architectural Advisor is in a position to assist users in decisionmaking and in establishing official system policies that constrain planned activities.

8.2.4. Strategic Planner

An agent in the Strategic Planner role proposes configurations to the agent system regarding where power and heat is generated, transmitted, and used; and proposes contingency plans.

8.2.5. Strategic Monitor

An agent in the Strategic Monitor role tracks system-wide status and identifies when the system is and is not operating according to the agreed-upon configuration.

8.2.6. Tactical Monitor and Controller

An agent in the Tactical Monitor and Controller role is responsible for making tactical control decisions in support of the strategic plan and is responsible for giving advance notice of actions to the strategic monitors.

8.2.7. Device Protector

An agent in the Device Protector role monitors the state of a device and acts to prevent damage to the device, including configuring the device to act properly under contingencies. This includes taking it online and offline, negotiate down times, etc.

8.3. ABCDIR Agent System Liveness and Safety Conditions

1. Human interaction
    a. When a person with the appropriate authority issues a command that the system is able to obey, the system should obey the command and report that it has done so.
    b. When a person with the appropriate authority issues a command that the system cannot obey, the system should report that it cannot obey the command and say why.
2. Source control
    a. When total load is in excess of the maximum that the system can supply in its current configuration, transition to a new configuration, if any, that can supply adequate power. Check first for stored configurations designated as capable for equivalent loads.
    b. If there are no appropriate stored configurations, search for some.
    c. When choosing a configuration to supply power, prefer configurations that:
        i. Generate equivalent power at lower cost;
        ii. Differ less from the preceding configuration;
        iii. Have a lower system-wide average fraction of rated power being carried by all lines
        iv. Can supply a thermal load that occurs within the appropriate time interval
    d. If it appears that total load will at some future time exceed the maximum that the system can supply in its current configuration, search for other configurations in which the projected load can be satisfied. Record each such configuration in conjunction with associated load information and other information needed to select among configurations. Denote the configuration as capable of satisfying its associated load.
3. Load control
    a. Maintain service to all loads.
    b. When load must be shed, shed non-critical loads before critical loads.
    c. Prefer supplying critical loads to shutting down sources for maintenance.
    d. Prefer shutting down sources for maintenance to supplying non-critical loads.
    e. If it appears that projected load will soon be greater than the system can supply, determine the order in which to shed loads and which loads should be shed.
4. Maintenance scheduling
    a. Take components offline as required by their maintenance schedules.
    b. As a component nears 90% of its MTBF, assign it high priority for being taken offline.
    c. Take any component that exceeds 90% of its MTBF offline for maintenance (may be overridden by 3c).
5. Distribution path
    a. When a distribution path fails, compute the power flow for the remaining network and determine whether any of the remaining lines will be forced to carry more power than their rated capacities.
    b. If a line is carrying more than its rated capacity, and there exists some other line not in service whose placement into service will allow a new load distribution where no lines are overloaded, place that line into service. If more than one such line exists, choose the line for which the system-wide average fraction of rated power being carried by all lines is lowest when the system is placed in the suggested configuration.
    c. When the steady-state power-flow through any generation, transmission, or distribution element exceeds its maximum steady-state rating, transition to a new configuration that will bring all system power-flows into specification. Such reconfiguration may include a prioritized shedding of heat loads or those that can be served by other heat sources.

8.4. ABCDIR Liveness Expressions

The ABCDIR agent system is largely asynchronous, so its agents should be able to deal with any message at any time. Liveness and safety constraints do not, therefore, lend structure to interactions. The "x^{ω}" notation means "x occurs indefinitely often" and is the only form of liveness expression needed (See Appendix 2: Formal notation for liveness expressions for other forms). It is somewhat pedantic to re-list the entire set of messages with the x^{ω} notation, but it is necessary to consider each message from the point of view that it could be received an arbitrary number of times, at arbitrary times, by any ABCDIR agent.

<u>User</u>
- (open GUI request)$^{\omega}$
- (query)$^{\omega}$
- (change request)$^{\omega}$
- (action confirmation)$^{\omega}$
- (device request)$^{\omega}$
- (panic button)$^{\omega}$

<u>System POC</u>
- (GUI)$^{\omega}$
- (network representation)$^{\omega}$
- (advice & rationale)$^{\omega}$
- (policy assessment)$^{\omega}$
- (consequences)$^{\omega}$
- (request confirmation)$^{\omega}$
- (update)$^{\omega}$
- (report)$^{\omega}$
- (query)$^{\omega}$
- (change request)$^{\omega}$

<u>Strategic planner</u>
- (consequences)$^{\omega}$
- (confirm action)$^{\omega}$
- (request report)$^{\omega}$
- (strategic update)$^{\omega}$
- (hypothetical response)$^{\omega}$
- (strategic monitoring info)$^{\omega}$
- (local monitoring info)$^{\omega}$
- (control info)$^{\omega}$

<u>Architectural advisor</u>
- (advice & rationale)$^{\omega}$
- (policy assessment)$^{\omega}$
- (hypothetical scenario)$^{\omega}$
- (policy adjustment)$^{\omega}$

<u>Strategic monitor</u>
- (network structure)$^{\omega}$
- (problem report)$^{\omega}$
- (strategic report)$^{\omega}$
- (strategic projection)$^{\omega}$
- (strategic update)$^{\omega}$

<u>Tactical monitor and controller</u>
- (network structure)$^{\omega}$
- (problem report)$^{\omega}$
- (tactical update)$^{\omega}$
- (tactical report)$^{\omega}$
- (device configuration)$^{\omega}$
- (state info)$^{\omega}$
- (contincency plan)$^{\omega}$

<u>Device protector</u>
- (device configuration)$^{\omega}$
- (state info)$^{\omega}$
- (diagnostics)$^{\omega}$
- (network structure)$^{\omega}$
- (problem report)$^{\omega}$
- (performance report)$^{\omega}$

8.5. Power System Simulation

The simulation is the implementation of the environment. As such, the simulation provides the APIs of the components of the environment with which the agent is expected to interact. See Section 7, "Modeling the Four-tent Microgrid", page 35.

9. ABCDIR Role Interactions

Agents need to interact with one another to accomplish power management objectives. Section 9.1 enumerates and describes all interactions needed to effect distributed power system management allocated to the roles described in Section 8.2, grouped by sending role. For each role, the messages sent by each role are named and described. A description is given for only the first appearance of a message; no description is given for the second and subsequent appearances of a message. There is no entry if a role doesn't send messages to a given role. Table 1 shows the same information without the descriptions. Section 9.2 describes data elements that need to exist in order to send the messages described in Section 9.1. Sending, receiving, and responding to these interactions is the means by which the agents accomplish the business of the agent system. Developing the agent system would consist in large part in implementing hardware and software capable of forming, sending, receiving, and behaving appropriately with respect to these messages.

9.1. Interaction Messages and Their Definitions

9.1.1. From User:

<u>*to System POC*</u>

Open GUI request	request to open a portal for communication with a system agent
Query	request for a future state prediction given hypothetical configuration(s)
Change request	request to change the current system configuration to an alternative state
Action confirmation	confirmation to proceed with a stated action

<u>*to Device Protector*</u>

Device request	request a status report (i.e. view the current state or trend information) or change of state (i.e. shut down or turn on device) directly from a device
Panic button	emergency shutdown command

9.1.2. From Architectural Advisor:

<u>*to System POC*</u>

Advice & rationale
Policy assessment

<u>*to Strategic Planner*</u>

Hypothetical scenario	an admissible system plan: an initial configuration, a final configuration, a set of constraints, and a set of event/time pairs
Policy adjustment	a new policy, including an initial configuration, a final configuration, a set of constraints, and a set of event/time pairs

9.1.3. From System POC:

to User

GUI	a portal for user agent communication
Network representation	the current system configuration and status viewed by the user in the GUI
Advice & rationale	a suggested system plan and likely change impact
Policy assessment	a suggested system policy and likely change impact
Consequences	the impact of a change request
Request confirmation	request user to confirm an action
Update(s)	update of the current policy and strategic plan
Report(s)	update of request completion, network problems, and current network trends

to Architectural Advisor

Query

to Strategic Planner

Change request
Request confirmation

9.1.4. From Strategic Planner:

to System POC

Consequences	
Confirm action	
Request report	request updated status of request completion
Strategic update	update of the current policy and strategic plan

to Architectural Advisor

Hypothetical responses	expected results corresponding to a set of hypothetical scenarios

to Strategic Monitor

Strategic monitoring information	comprehensive monitoring information for broad network observation

to Tactical Monitor & Controller

Local monitoring information	monitoring information for local device supervision
Control information	execution information for local device control

9.1.5. From Strategic Monitor:

to System POC

Network structure	
Problem report	update of network problems/alarms
Strategic report	update of current network trends

to Architectural Advisor

Network structure	
Strategic projection	anticipated network state based on trend information

to Strategic Planner

Network structure
Strategic update

9.1.6. From Tactical Monitor & Controller:

to Strategic Monitor

Network structure	
Problem report	the tactical managers notify one another of problems
Tactical update	report of local problem situations and measures taken
Tactical report	update of local device performance

to Tactical Monitor & Controller

Problem report

to Device Protector

Device configuration	state of device settings (i.e. on/off, input/output level, etc.)
State information	state of device (i.e. on/off, input/output level, etc.)
Contingency plan	prescribed actions for unexpected circumstances

9.1.7. From Device Protector:

to User

Device configuration	state of device settings (i.e. on/off, input/output level, etc.)
State information	state of device (i.e. on/off, input/output level, etc.)
Diagnostics	component readouts (i.e. hours since startup, exhaust temp)

to Tactical Monitor & Controller

Network structure
Problem report
Performance report

Table 1: Interactions between Roles

This table describes the information that passes between agents in each role. The contents were derived from the protocol definitions. The agent in the left column sends messages of the type appearing in a table cell to the agent at the head of the column.

to ↑ / from →	**User**	**to System POC**	**to Architectural Advisor**	**to Strategic Planner**	**to Strategic Monitor**	**to Tactical Monitor & Controller**	**to Device Protector**
from User		Open GUI request, Query, Change request, Action confirmation					Device Request, Panic Button
from System POC	GUI, Network Representation, Advice & Rationale, Policy assessment, Consequences, Request confirmation, Updates(s), Report(s)		Query	Change request, Request confirmation			
from Architectural Advisor		Advice & Rationale, Policy assessment		Hypothetical scenario, Policy adjustment			
from Strategic Planner		Consequences, Confirm action, Request report, Strategic update	Hypothetical response		Strategic Monitoring Information	Local monitoring information, Control information	
from Strategic Monitor		Network structure, Problem report, Strategic report	Network structure, Strategic projection	Network structure, Strategic update			
from Tactical Monitor & Controller					Network structure, Problem report, Tactical update, Tactical report	Problem report	Device configuration, State information, Contingency plan
from Device Protector	Device configuration, State information, Diagnostics					Network structure, Problem report, Performance report	

9.2. Data Class Definitions

This section contains design information about the data that needs to be passed between agents in different roles during the interactions declared above. This section captures what needs to be represented; the implementation details of class hierarchy, stored vs. computed values, explicitly or implicitly held data, etc. are deferred.

The difference between the things defined here and the things defined in Section 8.1, ABCDIR Environment, is that the Environment section describes the properties of the objects that the multi-agent system will *interact with*, while this section describes the logical environment and additional supporting data objects the system must *represent*. Things in the environment are represented in the system in nearly every case, so this section (from which object class definitions specs are largely derived) captures everything in the Environment.

9.2.1. Devices

Models of the physical components of the environment are used to express the state of the environment, pose planning problems, and reason about the effects of actions. Loads and sources can be either power or thermal (sometimes both), but transmission always refers to power, since heat cannot be transported over meaningful distances and so must either be used at the generation site or discarded.

Sources (Generators)
- Make/Model
- Fuel (gas/oil, coal, wind, water, etc.)
- State (on/off/spinning)
- Production Rate (real & reactive)
- Thermal Output
- Capacity (minimum & maximum)
- Associated Storage
- Set Point and Droop Rate
- Location
- Performance Metrics
- Heat Rate Curve
- Reaction Rate
- Response Time
- MTBF
- Time Since Last Maintained
- Uptime

Loads
- Demand (real & reactive)
- Demand History
- Demand Forecast
- Thermal Load
- Priority
- Aggregation Information
- Location

Switches
- State (on/off)
- Location

Lines
- Current
- Impedance
- Power (real & reactive) – each way
- Capacity (maximum)
- Type (transmission or distribution)
- Length
- Line Cost
- Temperature
- Location

Buses
- Voltage
- Magnitude
- Angle
- Capacity (voltage min & max)
- Type (generator – regulates V or Q, load, or swing)
- Location

Transformers
- Scale (i.e., input vs. output voltage)
- Location

Sensors
- Sensors may or may not be represented internally, depending on whether the agents need to reason about properties of the sensor itself; e.g., its state of health.

9.2.2. Power Networks

To reason about the environment as a system, we must have a means of representing the logical and physical structure of the network that contains all of the devices in the environment. Above a certain level of abstraction the details of each section of the network will be less important than the externally visible aggregate properties of those sections.

Physical Power Network

- Physical network topology
  - Grid Input (internal & external sources)
  - Transmission/Distribution Network Topology
  - Grid Output (internal & external customers, storage, ground)
- Aggregate statistics, performance metrics, etc.
- Network priorities

Logical Cell Network

- Logical network topology
- Aggregate statistics, performance metrics, etc.
- Introspective details and network priorities
  - Economic Factors
  - Network Safety
  - Redundancy
- Cell parameters – net and gross consumption and production

Configuration

A network or set of networks or set of devices and their states. A configuration is the data structure that represents the states of the devices that make up the network(s).

Forecast

A predicted configuration based on a real-world configuration

9.2.3. Events

Event One of: device creation, device deletion, load increase, load decrease, transmission line open, switch open, switch close, source halt, source start, source max output change

Events can be either expected or unexpected; events that are controllable can be either intended or unintended. An unexpected event cannot be intended. Intended events are always expected.

Device(s) Involved

- Previous Device State
- Current Device State

9.2.4. Plans

Plan A scenario whose configurations are complete and consistent and whose actions are executable by the agents. The "plan" is that, for each configuration/time pair, the agents will cause the system to be in the given configuration at the given time by executing the associated (i.e., preceding) actions.]

Planning domain

Planning problems

Planning constraints (aka Policies)
Economic Factors
Reliability Factors
Safety Factors

9.2.5. Messages

Many message types are specializations of more abstract concepts; e.g., a query for data and a request for service are both interrogatives. Messages normally contain information denoting their types. Whether the information that a given message is of a particular type is implicit, based on its content, or explicit, based on, e.g., its container, is up to the implementation. See Section 8.4 for a list of messages used within the system and Table 1 (p. 62) for the senders and receivers.

Load profile A set of ordered pairs, each a load followed by a time. The ordered pairs are ordinarily considered in temporal sequence. Given two such ordered pairs, between which (temporally speaking) there is no third pair, the load at any time between them is considered to be the load at the earlier time.

A load profile can be considered a record, a prediction, or an estimate.

Scenario A series of ordered pairs, each consisting of a configuration followed by a time, and the actions or events, if any, that cause the transition from each configuration to the next one in temporal sequence.

— This page intentionally left blank —

10. ABCDIR Agent System Protocols

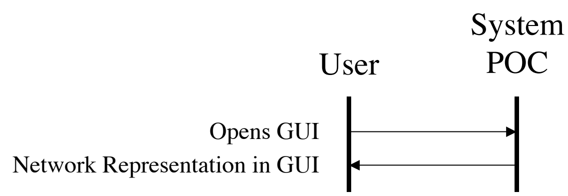
A *protocol* is a collection of send/receive actions bundled together in the name of the objective the protocol is intended to achieve. This section describes the desired behavior of the initial sender (named for each protocol under "Initiated by:") and subsequent recipients ("partners") for the mesages described in Section 9. Protocols at the tactical level are stated in the singular voice for simplicity but may involve multiple instantiations; e.g., there may be multiple Device Protectors.

10.1. Start User Interface

Performed when a user first contacts the System POC during any session.

Initiated by:	User
Partners include:	System POC
Input:	User Contact
Output:	Open GUI

The user approaches a System POC terminal, either physically or over the network. The System POC presents the system configuration and network status, and can provide reviews of recent network history (problems and resolutions, trends, etc.).

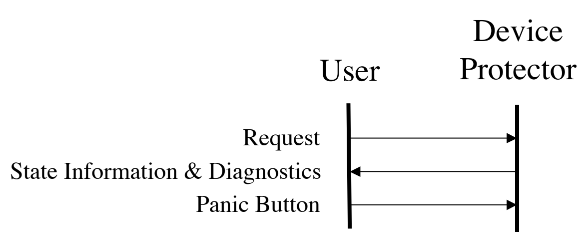


10.2. Device Interface

Performed when a user needs to directly interact with a component of the system, either under nominal or emergency circumstances.

Initiated by:	User
Partners include:	Device Protector
Input:	User Contact
Output:	Device response

The user approaches a device physically and interacts with it through any interface available. This may result in the user accessing diagnostic information about the device, shutting the device down for maintenance, or forcibly taking an unresponsive device offline. Either the device itself or other components of the system may initiate other protocols (not shown) so that the rest of the system will respond appropriately to the device's new state.

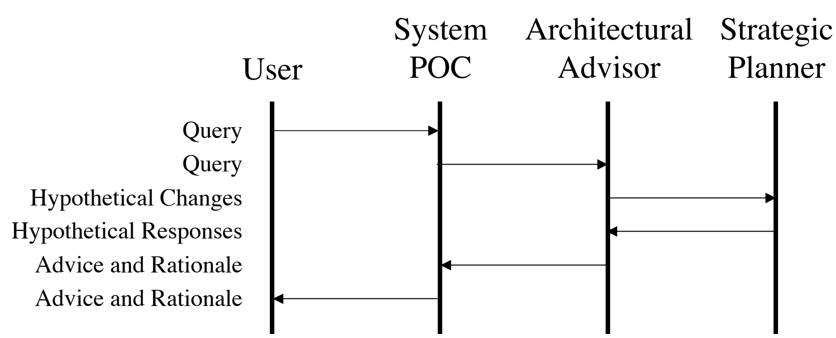


10.3. Submit User Query

Performed when a user wants to deliberate among various potential changes.

Initiated by:	User
Partners include:	System POC, Architectural Advisor, Strategic Planner
Input:	Proposed under-constrained configuration change to one or more existing and/or newly added devices.
Output:	Audit log of the request, report and rationale of recommended fully constrained configuration change.

When a User would like to know how best to change the way the system is operating, the User initiates contact with the System POC. The User may describe abstract changes to the system, which are interpreted as a question of how to best finish specifying the changes. As queries are made, the System POC logs them and passes them on to the Architectural Advisor. The Architectural Advisor generates more complete specifications, passes these to the Strategic Planner, and gets corresponding system plans back. The Architectural Advisor evaluates the expected results against one another, and the best configurations are returned to the User via the System POC along with the likely impact of changes. The preferred changes may be presented to the User so that a specific request can be made based on the recommendations.



10.4. Submit User Request

Performed when a user wants to add, remove, or adjust components in the system.

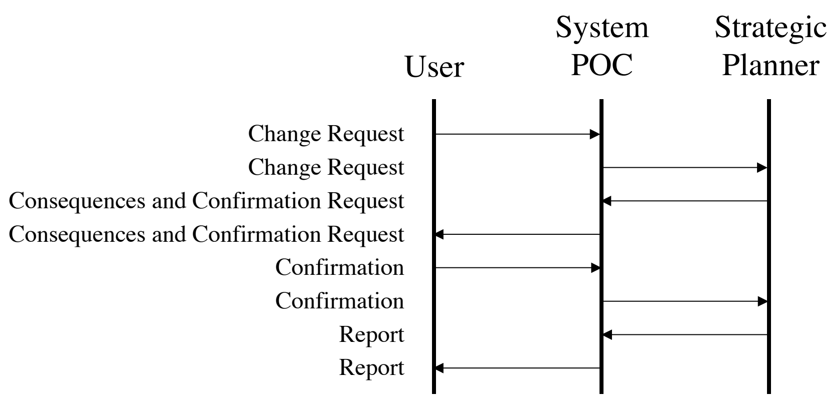
Initiated by: User

Partners include: System POC, Strategic Planner

Input: Proposed configuration change to one or more existing and/or newly added devices.

Output: Audit log of the request, report and rationale of request being accepted or rejected, and (if accepted) a new goal for the Strategic Planner.

When a User would like to change the way the system is operating, the Users initiate contact with the System POC and may make modifications to the system as presented by the POC. When the user is satisfied with the set of modifications, the System POC will log and pass this configuration to the Strategic Planner. If a plan can be prepared that both satisfies plan goals in the context of the current network and is within current system policy, the request will be accepted, subject to the authority level of the User. The Strategic Planner returns the plan and the projected consequences of executing the plan to the System POC. The System POC displays these to inform the User of the likely impact of changes. The System POC then asks the User whether to proceed with the plan or not. If the User chooses to proceed, the Strategic Planner forms a goal to enact the new plan. If not, the goal and plan are discarded. Future consideration may choose to archive the goal, plan, and relevant system states for future use, but this is not central to the protocol and is not shown.

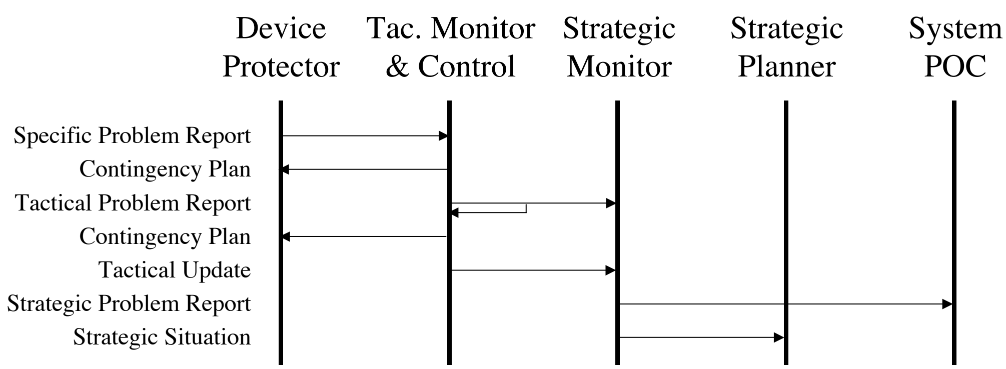


10.5. Raise Alarm

Performed when system contingencies occur (protection faults, etc.) or when trends in the system are expected to lead to contingencies or to violate safety conditions.

Initiated by:	Strategic Monitor, Tactical Monitor and Controller, or Device Protector
Partners include:	System POC, Strategic Planner
Input:	Problem description
Output:	Increased system awareness, problem report, initiated contingency plans, agent goals.

When local protection-relevant conditions begin to deteriorate (or precipitously cross some threshold) the Device Protector's task is to notify the Tactical Monitor and Controller (TMC) of the situation, and what measures will be or have been taken to provide immediate system protection. The TMC collects this information from its tactical region and considers it with the continuous tactical view. If the TMC can deal with the situation locally without violating strategic requirements, it may initiate pre-arranged contingency plans or compute and initiate new tactical plans; in addition it must report the situation and the enacted response to the Strategic Monitor and its neighboring TMCs. If the TMC cannot isolate the event, it may take local actions to mitigate the strategic impact, and it must report the situation and any local measures to the Strategic Monitor. The Strategic Monitor must forward this information to the System POC and the Strategic Planner. The Strategic Planner must take the new system status and projections and form a goal to generate and distribute a new strategic plan. Until a new plan is distributed the TMCs may initiate strategically directed contingency plans and must keep the Strategic Monitor informed of developments.

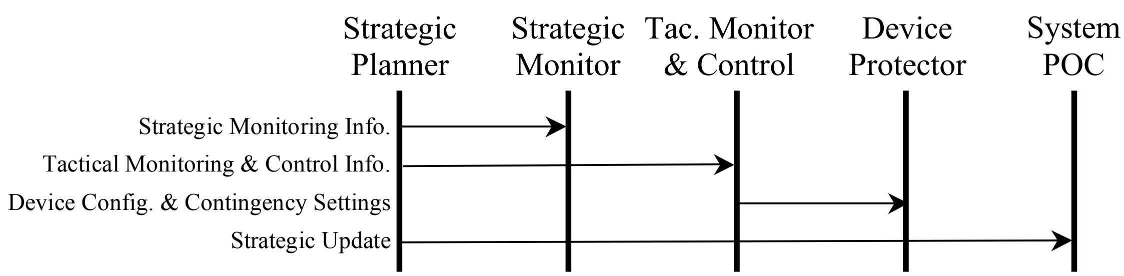


10.6. Distribute Plans

Performed when a new plan (see 9.2.4) is to be adopted.

Initiated by:	Strategic Planner
Partners include:	Tactical Monitor and Controller, Strategic Monitor, Device Protector, System POC
Input:	Planning Goal
Output:	Strategic, tactical, and contingency plans.

When the Strategic Planner receives a goal to generate a new plan by user request or an internal alarm, its task is to evaluate the current system status and formulate a new strategic plan. When the new plan is complete, the Strategic Planner forwards it to the System POC and distributes it to Strategic Monitor, the Tactical Monitor and Controller (TMC), and Device Protector, as follows: The Strategic Planner sends strategic monitoring information to the Strategic Monitor for broad network observation. The Strategic Planner sends The TMC execution and monitoring information for local device control and supervision. This tactical information is then filtered down from the TMC to the Device Protector as a new configuration state and contingency settings.

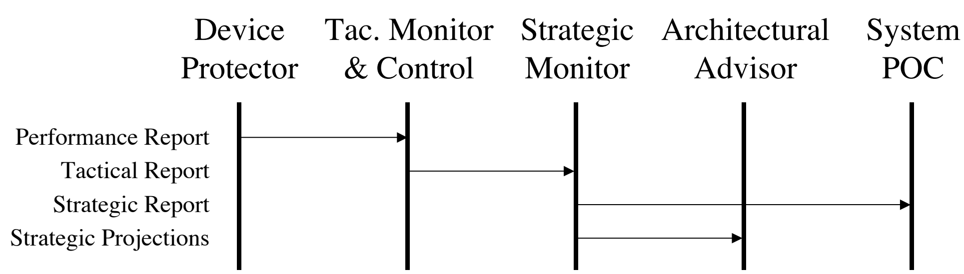


10.7. Report Trends

Performed continuously to maintain situational awareness, at both tactical and strategic levels, of system behavior and changes in behavioral patterns.

Initiated by:	Strategic Monitor, Tactical Monitor and Controller, or Device Protector
Partners include:	System POC, Architectural Advisor
Input:	System Report
Output:	Increased system awareness, performance logs, and projections of trends.

System conditions are of interest at all levels, and deviations from the currently recognized system patterns are reported "upward" even if no immediate problems are apparent (if problems *are* anticipated, an alarm is raised instead – see Section 10.5, "Raise Alarm"). The Tactical Monitor & Controller collects and reviews performance reports from Device Protectors and combines them into tactical reports, which it sends to the Strategic Monitor. The TMC raises an alarm if it identifies problems at the tactical level; the Strategic Monitor raises an alarm if it identifies problems at the strategic level. In any case, the Strategic Monitor combines the tactical-level reports into a system-wide report, which it forwards to the System POC, who records it. The Strategic Monitor also develops strategic projections based on trends in the reported information and sends this to the Architectural Advisor so operational policies can be evaluated against the projections.

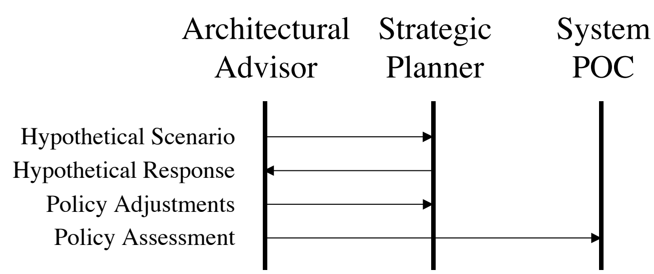


10.8. Assess Policy

Performed by the Architectural Advisor to re-evaluate existing operational policies in light of reported system trends.

Initiated by:	Architectural Advisor
Partners include:	Strategic Planner, System POC
Input:	Strategic Projections
Output:	Policy Reviews, Policy Changes

When the Architectural Advisor receives trend information in the form of strategic projections, its task is to evaluate the likelihood of the trends continuing to varying degrees beyond nominal system performance. The Architectural Advisor does this by generating hypothetical scenarios based on the current system and these projections. The Architectural Advisor then requests operational plans from the Strategic Planner to describe the system's hypothetical response to those situations. The Architectural Advisor compares different policies by evaluating the hypothetical responses for the same scenario under the different policies. The policies are compared to one another based on the utility of the system's response and the likelihood that the scenarios will occur.

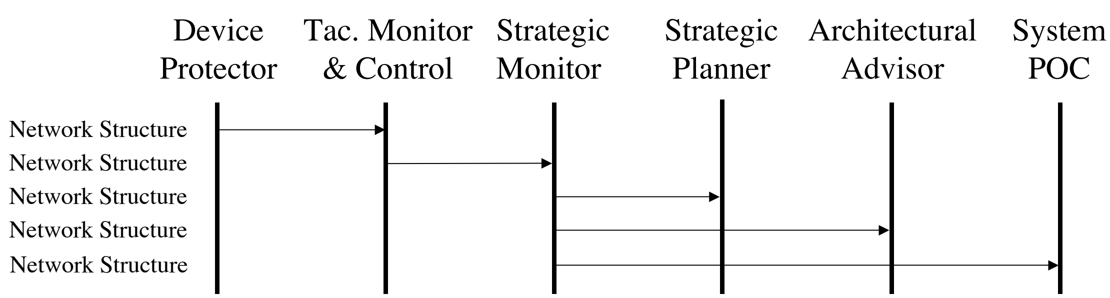


10.9. Plug and Play (P&P) Support

Performed by the monitoring systems when new components are added to the network (not when known components are enabled and disabled). This protocol is how the data structures that represent the network are updated to reflect physical changes in the network.

Initiated by:	Device Protector
Partners include:	Tactical Monitor and Controller, Strategic Monitor, Architectural Advisor, Strategic Planner, System POC.
Input:	New device
Output:	Network structure

When a new device is plugged into the system, or when an existing device is returned to the system after having been completely shut down and effectively removed from the system, the Device Protector for the device must report the existence of the device to the tactical Monitor & Controller, who in turn reports the new device to the Strategic Monitor. The Strategic Monitor informs the Architectural Advisor, Strategic Planner, and System POC of the changes so new policies and plans incorporating the new device can be generated.



11. ABCDIR Agent System Organizations

In this section, we give the rationale for agent classes, roles, and faces, which are the organizing principles for constructing the agents needed for an application. These are then related to cell, glob, and co-op organizations. See Section 4.1, "Organizing Principles for Agents Managing a Network," for the principles that motivate Cells, Globs, and Co-ops.

An *agent* is a locus of functional elements. An agent *class* can be thought of as a template for producing agents of a particular type. The motivation for the agent class as an organizing principle is to group several faces together in such a way that agents with the capabilities needed to assume those faces can be readily constructed. John Doe, a named individual having the job of doorman for the Ritz-Carlton Hotel on Central Park in New York City, is an agent. The fictional class "Doorman" would be one of perhaps several template classes used in the "construction" of Mr. Doe.

A *face* is a collection of related roles associated with some responsibility to the collective. Initially, we assigned roles directly to agent classes. As we proceeded with the Gaia analysis, we realized that identical collections of roles were being assigned to different classes of agents, implying a missing organizing principle, which we call a "face." A face is defined by the messages it can decide to send, the messages it must be able to receive, and the nature of its responses to messages it receives. The motivation for a face is to be able to name collections of related roles for assignment to different agent classes. The "Entry monitor and controller" role is part of the "Building security" face, and the "resident entry/exit assistance" role is part of the "external POC" face.

A *role* is a collection of functional elements; if an agent is capable of performing all the functions, it can assume the role. The motivation for a role is to be able to identify a set of related activities as being the responsibility of agents assuming a particular face. "Entry monitor and controller" and "resident entry/exit assistance" are roles that a doorman might play, depending on whether he had assumed the "building security" face or the "external POC" face, respectively.[13]

11.1. Cell Organization

A cell is a set of power sources, loads, and their connective network, the combination of which is simple enough to be managed by a single entity based on largely local principles. Figure 23 shows an example of a cell.

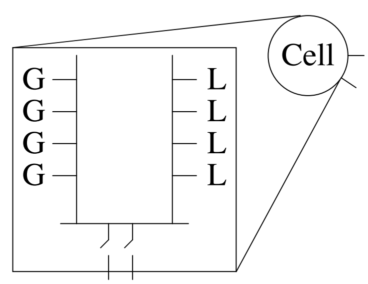


Figure 23: Cell organization (G = generator, L = load)

11.1.1. Cell organizational rules

A cell must generate enough power to meet its loads in the absence of conditions that dictate other behavior. If an independent cell generates more power than its loads demand, it must store or somehow discard the excess, and if it cannot generate enough power to satisfy cell loads, some load must be shed.

[13] Agents of classes other than the "Doorman" class might also be able to fulfill these roles.

11.1.2. System roles exhibited by agent faces in a cell

The data and scope of concern for protocol execution in the cell is within-cell only.

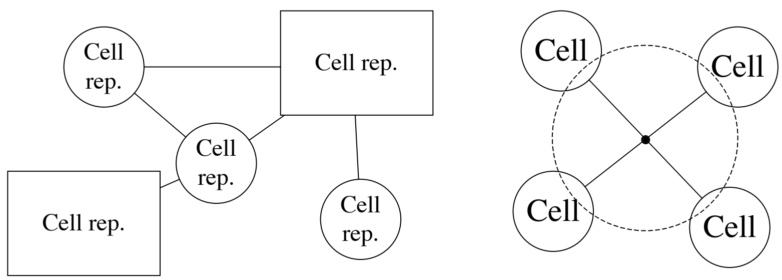
The *Cell Operator* face performs the protocols for Strategic Monitor, Strategic Planner, and Tactical Monitoring and Control (Start user interface, Submit user request, Raise alarm, Distribute plan, Report trends) in order to plan operations and execute its plans.

The *Cell Logistics* face internally performs the protocols for System POC, Architectural Advisor and Strategic Planner (Start user interface, Submit user query, Assess policy) in order to develop internal policies.

The *Protection System* face internally performs the protocols for Device Protector (Device interface, Raise alarm, Report trends, Plug & Play support).

11.2. Glob Organization

As described above, a Glob is a generalized graph of Cells (either simple cells or co-ops interacting as cells), where the graph represents elements of the physical power network. Cells within a glob can trade power to either make up a deficit or dispose of a surplus. Cells in a glob need to accommodate all the functions of basic cells.



(a) (b)

Figure 24: Glob organization

Figure 24 shows two different representative globs of cells (they could also be co-ops, since there is no indication of whether they are following a common policy).

11.2.1. Glob organizational rules

Cells in a glob must communicate with physically adjacent cells. It is somewhat more revealing to say that if two physically adjacent cells cannot communicate with one another, they cannot both be part of the same glob. See Figure 24 (a). Cells maintain communication connections based on the physical network, necessitating a parallel logical network (note that the physical network may include a potential power transport paths that happens to be cut by an open breaker).

Cells in a glob are independent. Glob cells are self-interested and negotiate as peers. Each cell is responsible for its own transmission commitments. Collaboration and information sharing are not prohibited, but if they are needed, a co-op should at least be considered.

Cells on a bus must manage the bus collectively. This is an exception to the "cells in a glob are independent" rule above. All ends of an n-ary arc (i.e., a bus) in the physical network must respond to the authority of the group of cells connected by the arc (see Figure 24 (b)). Voting, DAG systems, special-purpose breakers, etc. are options for enforcing this, as would be litigation against cells that did not behave properly in response to group decisions.

11.2.2. System roles exhibited by agent faces in a glob

An agent operating a cell that's part of a glob needs to be able to do everything an agent operating a standalone cell can do. The scope of concern for these "glob agents" is expanded to include the existence of other entities able to provide and accept power. This new responsibility is encapsulated in the Cell Representative. In general, however, since glob agents have no responsibilities outside the cell, their relationships with the power system and users are not changed much from those of a standalone cell agent.

The *Cell Representative* is an agent that can publicly perform the appropriate protocols related to System POC, Strategic Monitor, and Strategic Planner (Submit user[14] query, Distribute Plan, Report trends, Raise Alarm). The System POC allows general interaction with the cell and its components from across the network, subject to authorization; the Strategic Monitor allows a cell to communicate its status to its neighbors; and the Strategic Planner is responsible for negotiating transmission in or out along each arc from the cell.

11.3. Co-op Organization

The co-op looks externally like a cell: Contact with the co-op is through a designated point of contact (POC), through which one negotiates to do business. The implementation should be constrained so that co-op POCs and cell POCs look the same.

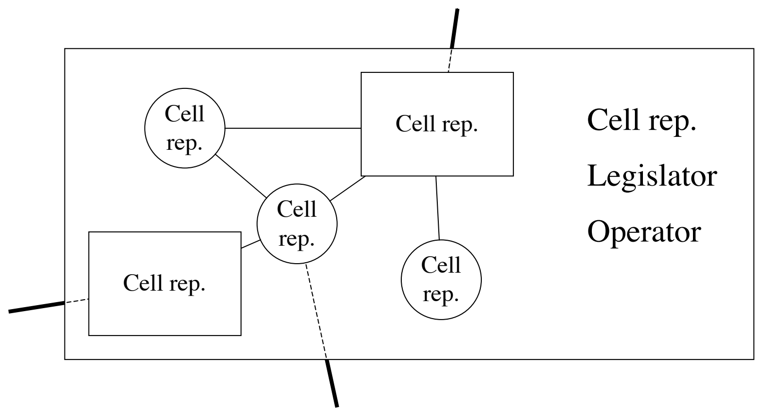


Figure 25: Co-op Organization

The co-op can be implemented in a variety of ways: the negotiating cell could have the authority of the cell rep, or could be the "ambassador" for collective negotiations being conducted internally, or could be a man-in-the-middle discussing things with each interested cell, transferring negotiations to the cells on those specific lines while still constraining and provisioning the negotiations and staying informed of and signing off on the resulting obligations.

Figure 25 shows a co-op with its external agent faces named.

11.3.1. Co-op organizational rules

The co-op is externally equivalent to a cell. The public face of a co-op is the same as the public face of a cell. Negotiations about lines in or out of the co-op must be taken up with the representative for the co-op.

Cells in a co-op must communicate with physically adjacent cells (as in a glob) and with the co-op organization. Cells of a co-op must communicate with whatever real and/or virtual entities govern the co-op. The governing instrument of a co-op is the policy statement.

[14] This message might be better named "Submit query" (i.e., no "user"), because its content is "what's the outcome of this configuration", interpreted in the present context as including "what would you say if I asked you for this much power?" and "Are you going to have a need for this much power?"

Co-op needs take precedence over individual needs. Cells in a co-op must operate according to co-op policies and plans. This is what makes a glob into a co-op.

Co-op policies may allow individual and collective freedom. To the extent that the governing body's decisions under-constrain the behavior of the individual co-op elements, the elements may internally meet co-op and any other private requirements in any compatible way. Glob organizational rules should apply[15] where not superceded by co-op rules, to allow the co-op components to negotiate freely when not bound by co-op policy.

The co-op may require internal knowledge of its elements. The governing body must remain sufficiently aware of the internal makeup of the co-op member cells to enable it to represent and reason about policy and plans at a suitable strategic level. Reciprocally, member cells must inform the governing body about, for example, critical and noncritical loads (or critical/noncritical buses that may have varying loads).

11.3.2. System roles exhibited by agent faces in a co-op

The *Co-op Representative* is an entity[16] that can publicly perform the appropriate protocols related to System POC, Strategic Monitor, and Strategic Planner *as they relate to the co-op* (These are identical with those laid out in 11.2.2 for the Cell Representative, that is, (Submit user[17] query, Distribute Plan, Report trends, Raise Alarm). The System POC allows general interaction with the co-op by other entities, subject to authorization; the Strategic Monitor allows the co-op to communicate its status to its neighbors; and the Strategic Planner is responsible for negotiating transmission in or out along each arc from the co-op. Co-op/Cell Representative is the face presented to the co-op governing entities—nominally, the Strategic planner (who governs operations) and the Architectural Advisor (who governs structure) internally by the cells within the co-op, whether they are simple cells or sub-co-ops presenting a cell rep interface.

The *Operator* face internally performs the roles of Strategic Monitor and Strategic Planner (Start user interface, Submit user request, Raise alarm, Distribute plan, Report trends) to establish plans and track collective operation. This is the co-op-level equivalent of the Cell Operator.

The *Legislator* face internally performs the roles of Architectural Advisor, Strategic Monitor, and Strategic Planner (Start user interface, Submit user query, Assess policy) in order to consider and form policy. This is the co-op-level equivalent of the Cell Logistics face.

---

[15] For rule-following purposes, the glob boundary is the boundary of the co-op of which the constituent elements are members.
[16] The term "entity" rather than "agent" is used here to allow the possibility that the cell rep could be a virtual entity whose authority derives from multi-party protocols.
[17] This message might be better named "Submit query" (i.e., no "user"), because its content is "what's the outcome of this configuration", interpreted in the present context as including "what would you say if I asked you for this much power?" and "Are you going to have a need for this much power?"

12. ABCDIR Agent Classes

The following agent classes define the multi-agent system that implements this design. Each agent embodies one or more faces. Each face requires the agent to fill one or more roles. The roles that are thus included in an agent determine the functionality that must be included in the agent. This functionality is implemented as extensions to the agent framework.

The agent classes are the kinds of agents that make up the system. Members of each class can participate in the processes for its faces (whether the processes are executed in a centralized or distributed fashion by several agents doesn't affect this).

Agent classes need to preserve a record of their faces, as these faces determine the scope of their authority (for example, an agent with the role of Strategic Planner in a Legislative face doesn't have the authority to order a plan be carried out, for example, since that is the Operator's responsibility – the Legislator only has planning capability to support deliberation).

Because the agents will all (ideally) be built upon a common agent framework, it would be nice to have faces declared and associated with certificates of authority and have the process functionality for each face be loaded at that time. Then, based on the faces, all the appropriate roles would be loaded to ensure that the agent has all the necessary functionality.

12.1. Cell agents

In a Cell, there are four faces that need to be represented by agents. Depending on the complexity of the system the protection system and operational details may or may not be separated into multiple agents, and then there is a higher-level management set of concerns that can be dealt with by a managerial agent running somewhat removed from the system.

12.1.1. Cell Manager agent

This agent manages the cell, and takes on the Cell Representative Face and Cell Logistics Face. This agent does not need to reside in the power system itself and may have substantial computational power available. It must communicate with the outside world (nominally via protocols that involve the User) and with the operational agents.

12.1.2. Protection System agent

This agent operates one or more devices, and is expected to be collocated with those devices. It takes on the Protection System Face.

12.1.3. Cell Operator agent

This agent operates the entire cell, at a mix of strategic and tactical levels. It takes on the Cell Operator Face. A "Facilities Supervisor" agent would combine the Protection System agent and the Cell operator agent into one agent corpus.

12.2. Glob agents

Globs are collections of cells, and the agents that can be cell representatives of a cell that belongs to the glob present that face for the cell. No additional class of agent is needed.

12.3. Co-op agents

12.3.1. Co-op Member Agent

This agent takes on the Cell Representative Face, Legislator Face, and Operator Face. The implication is that all Co-op Member Agents collaboratively perform the legislative and operational processes, and most likely a specific agent will be elected to perform the Cell Representative function on behalf of the Co-op, possibly deferring decisions to the collective or acting with the elected authority of the collective.

12.3.2. Co-op Manager Agent

This agent takes on the Cell Representative Face and Legislator Face. This agent (or a collective of these agents) makes policy decisions for and publicly represents the co-op.

12.3.3. Other Co-op Agents

Each of the Co-op Representative face, Co-op Legislator face, and Co-op Operator face would be represented by a class analogous to the similarly named cell agent (Cell Representative Agent, Cell Legislator Agent, Cell Operator Agent; See Section 11.3.2) and would serve the appropriate function either as a virtual entity (i.e., as a multi-party committee) or as an individual.

12.4. Trans-Organizational agents

Agent classes (however implemented) can be combined by a common subclass to create a realizable class for a new agent. Without enumerating them, this would be appropriate for creating an agent that has different class behavior within, for example, a Cell and its containing Co-op. One such agent might be a Cell Manager Agent from the point of view of the Cell and a Co-op Member Agent and Cell Representative from the point of view of the Co-op containing the cell.

13. Instantiating Agents and Roles for Four tents

Agents interact to ensure power is correctly supplied to and drawn from the common bus according to agent consensus. Each tent has its own power source, loads, and distribution network, allowing it to operate as a cell. A single agent will operate each of the four cells. The cell agents in this example will have three faces: cell representative, cell operator, and protection system. Roles and protocols supported by each face appear in Table 2 and are further described in the design document.

Table 2: The Roles and Protocols of Agent Faces

Agent Face(s)	Role(s)	Protocol(s)
Cell Representative	System POC Strategic Monitor Strategic Planner	Start User Interface Submit User Request Raise Alarm Distribute Plans Report Trends Plug & Play Support
Cell Operator	Strategic Monitor Strategic Planner Tactical Monitor & Control	Submit User Request Raise Alarm Distribute Plans Report Trends Plug & Play Support
Protection System	Device Protector	Device Interface Raise Alarm Distribute Plans Report Trends Plug & Play Support

The four cell agents manage the four tents as a single electrical system by cooperating in agreement with a shared priority, generation, and distribution policy. Co-op level "cell representative", legislator[18], and operator faces must also be implemented either as individual agents or domain authority groups (DAG) if they are needed in the system.

Possibility 1: pre-selected agent does all planning (is a Single Point Of Failure (SPOF), undesirable and contrary to system intent but adequate for demonstration)

Possibility 2: planning agent randomly elected for each problem (better but still a SPOF while plans are being generated)

Possibility 3: parallel state monitoring, state consensus, and parallel planning (the correct—i.e., distributed—way)

---

[18] The *legislator* face, which enables the agent to make policy decisions, is not necessary for the demonstration because the agents will operate according to a fixed policy. The Submit User Query and Assess Policy protocols need not be implemented nor the role of the Architectural Advisor in the Plug & Play Support and Report Trends protocols.

— This page intentionally left blank —

14. Agent Communication Network

To operate as a co-op, agents must be able to interact with each other through the exchange of objects across one or more networked hosts. Agents must also interact with exterior programs (i.e. Matlab) to monitor and affect simulated system state. Figure 26 is a graphical representation of the communication network.

Communication is required and assumed, and the performance of the ABCDIR system would almost certainly degrade, probably non-linearly, as the comm network degraded. Denial of network service by adversaries or natural causes is relevant and should be further examined.

We did not expend significant effort delineating the requirements of a network suitable for agent-based power system management, for the following reasons: First, the system has not been sufficiently realized to allow performance testing based on network degredation. Second, the communication network would most likely be either an existing private network or the Internet; we would suggest performance evaluation based on existing network models is probably more relevant than specifying the network requirements, although preliminary analysis to determine whether contemporary networks are in the ballpark for agent-based system management would be worthwhile. Finally, the high bandwidth, broad availability, and generally consistent level of Internet service imply that "common carrier" network service will be sufficient under normal conditions.

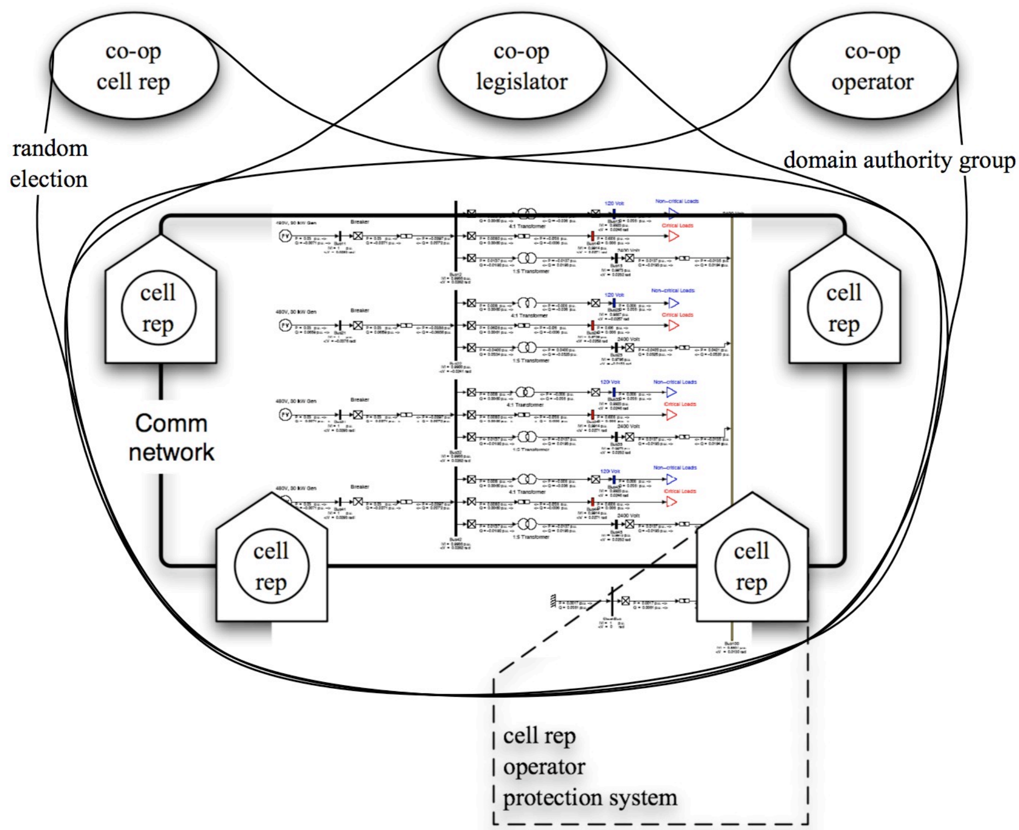


Figure 26: Agent Communication Network

— This page intentionally left blank —

15. Agent Implementation

15.1. Agent Framework Requirements to Support the Demonstration

This section lays out the minimum general requirements needed by an agent framework to realize the demonstration elements, agent roles, and plant models/analyses/simulations. An agent framework provider would match the capabilities of its agent framework to these requirements to determine whether the technology is a good match.

The four-tent microgrid network depicted in Figure 4 is the demonstration network.

Each of the four tents contains a generation source, a critical load, a non-critical load, a distribution network, and breakers to protect these elements. The power system of each tent constitutes a cell (see Section 4.3 for a discussion of this organizational approach) operated by an agent.

There is a shared distribution bus among the tents that enables the four cell agents to be organized into a co-op whose policy specifies priority, generation, and distribution. Essential policy specifies that all critical loads for all tents are to be served before any non-critical loads and that the collective load is to be satisfied by the collective capacity constrained by the distribution network.

1. The Cell Representative for the co-op is either randomly elected by the group or an agent designated to have that role.
2. The co-op Operator agent will be represented by a DAG comprising the four tent agents.
3. Strategic monitoring information for the virtual agent will be sent to the DAG with BB, consensus and plans will be done internal to the DAG in whatever manner is chosen, and group-signed messages from the DAG will be distributed as appropriate.
4. All agents will use a common ontology/domain model and be implemented somehow TBD [as extensions of the SAA2 framework atop AAF sites].
5. SAA2 API-compatible expectation/activation setup is required to support the existing schema mechanism.
6. ABCDIR protocols will be implemented as a set of schemata that each participating agent will execute.
7. Schema use and network monitoring will require expectation-driven operations.
8. Internal interaction among agents will occur by exchanging objects.
9. All messages must be signed by the sending agent or agents (as would be the case if the message were sent from DAG).
10. Sending a message to an agent should be a logical operation invoked by protocols (via schema system) using a target-appropriate method (signed message or BB schema).
11. Messages from an agent to a domain authority group (DAG) should use strong broadcast.
12. Agents must also interact with external programs and users for planning and reporting.
    a. Interaction with a Matlab model (via tcp/ip) is necessary to monitor and affect simulated system state.
    b. Agents must interact with an additional planning algorithm other than Matlab to determine set points and predict load flow.

c. Versions of these algorithms have been developed and can be used to support operational planning.

13. User interaction via HTTP is also necessary. Individual agents contacted by a user will forward the external user to the elected/dedicated cell rep for the co-op (no GUI to the cell reps of the individual tents will be provided for the demonstration or the GUI would have to be split and have a "co-op rep" link etc. etc.).

Requirements Note(s):

i. dclos message passing/perception, expectation driven reasoning, and the schema mechanism should suffice for the bulk of protocol implementation, including the refurbished crypto protocols

ii. http/html perception and expression should suffice to add a GUI on top of this

iii. shell program execution and processing may suffice to add planning internals if coordination/common-knowledge problems can be dispensed with

15.2. Ongoing Agent System Processes

15.2.1. Power Resource Allocation

The question "which sources should be making power, and how much?" requires not merely an answer, but constant updating. The general notion is that power should be generated and transported[19] in such a way that the total cost to provide power for a particular usage profile is minimized over some time interval. Consideration of short intervals (tens of minutes) leads to answers to the given question; lengthier time intervals (years to decades) can lead one to proper placement of sources.

In the early phases of the project, we understood that the primary ongoing task of an agent-based power management system would be allocation power production and transport resources. Although these tasks remain important, they're now perceived as being only fragments of what the agents need to do, as discussed in the remainder of this report. In any case, we expended some effort considering resource allocation, as discussed in the remainder of this section.

System Cost

Three main costs, start-up, generation, and transmission/distribution, should be considered when calculating a system cost. The start-up cost for a power source is a function of the fuel needed to bring it on line. Currently, start-up cost is included in the model for each power source, but does not affect the outcome because its value is assumed to be zero. These values can be updated with accurate data as necessary. Generation cost will be the largest contributor to the total system cost. This too is a function of the fuel consumed by each power source and in all practical cases, is a quadratic function of the power generated ([Saadat], p. 267). Equation 1 shows the general cost equation for power source *i* that is generating ***P*** kWs of power and has cost coefficients alpha, beta, and gamma.

$$C_i = \alpha_i + \beta_i P_i + \gamma_i P_i^2 \tag{1}$$

[19] We use the nontandard "transport" to describe the movement of power from one location to another to avoid the distinction between "transmission" and "distribution."

The total system generation cost is the sum of the individual cost functions associated with power sources each producing different amounts of power. Transmission/distribution cost (including the cost of generating the power lost during transmission) can be calculated but it is not currently included in the model due to the scale of the example.

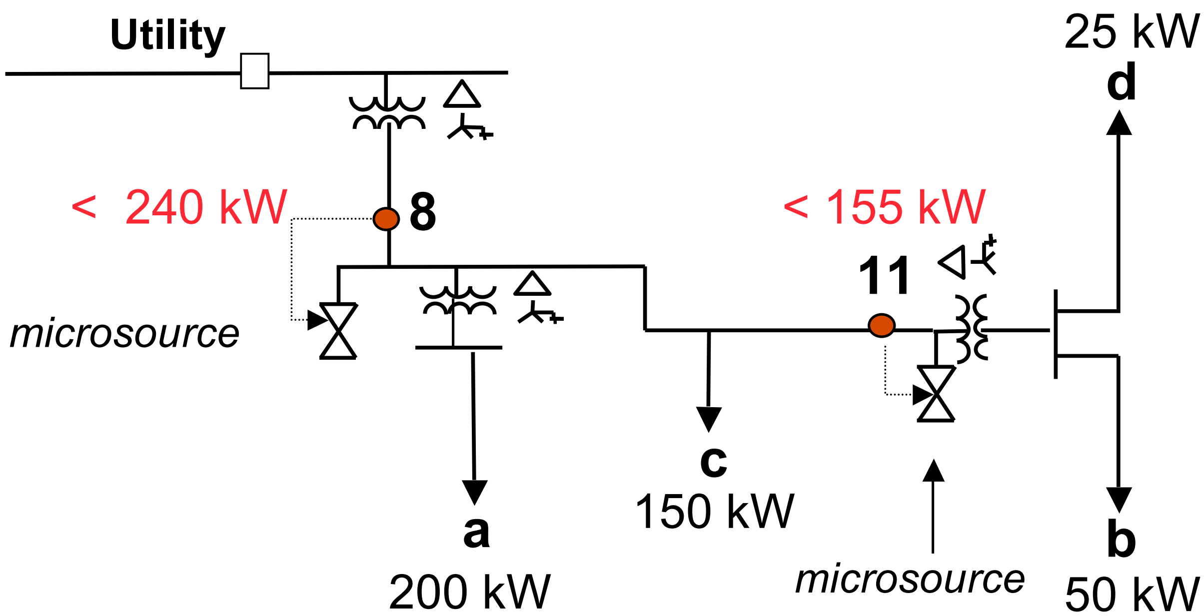


Figure 27: Microgrid for Optimization Example

<u>Optimization</u>

Notation

i	Power sources (utility, microsource 8, microsource 11)
j	Loads (a, b, c, d)

Parameters

$MinP_i$	The minimum power level power source *i* can produce when in operation • Note: If the source is not running, its power level will drop to zero.
$MaxP_i$	The maximum power source *i* can produce when in operation
CP8	240 kW, the maximum power that can flow through control point (CP) 8
CP11	155 kW, the maximum power that can flow through control point (CP) 11
$Load_j$	The amount of power demanded from site j

Variable

$P_{i,j}$	The amount of power supplied by power source *i* for load *j*

<u>Global Optimization</u>

Global optimization calculates the value of all system variables to ensure the best performance. In our case, we are trying to find the optimal power levels for each power source so that the total system cost is minimized and all loads are met. The global objective function, equation 2, is described below such that P_i is the total power generated at power source *i* to fulfill load demand at any site *j*.

$$\min \sum_i \alpha_i + \beta_i P_i + \gamma_i P_i^2 \qquad (2)$$

The following constraints are in place to ensure loads are serviced, power source capacity is maintained, and flow at the control points does not exceed set limitations. First, equation 3 ensures that demand at each site *j* is met by power generated from any combination of power sources.

$$\sum_i P_{i,j} \geq Load_j \forall j \qquad (3)$$

Although equation 3 suggests that the total power generated to meet the demand at *j* can exceed that which is required, the amount generated will never exceed demand since the objective is to minimize cost. The next constraint controls the generating capacity of each power source and is described in equation 4.

$$MinP_i On_i \leq \sum_j P_{i,j} \leq MaxP_i On_i \qquad (4)$$

As noted before, if a power supplier is in operation, it must follow its minimum and maximum capacity constraints. However, if the power source is not currently operating, the power level will be zero. To ensure this happens, a binary variable, On_i, is included. On_i will be zero if power source *i* is not in operation and one if it is. Equations 5 – 8 control the flow going through control points 8 and 11.

$$P_{Utility} \leq CP8 \qquad (5)$$

$$P_{MS8} + P_{MS11} - \sum_j Load_j \leq CP8 \qquad (6)$$

$$P_{Utility} + P_{MS8} - Load_a - Load_c \leq CP11 \qquad (7)$$

$$P_{MS11} - Load_b - Load_d \leq CP11 \qquad (8)$$

Equation 5 controls the power entering from the utility into control point 8. This will force the maximum power generated by the utility to be less than or equal to the control point 8, limit rather than the utility's capacity constraint. On the opposite side of control point are the microsources and electrical loads of the microgrid. The flow that is generated by the microsources and not consumed by the loads is controlled by equation 6. Again, since the goal is to minimize cost, this flow should be less than or equal to zero, unless the utility is willing to pay for the additional microsource power to stabilize its own generation pattern. Equations 7 and 8 control each side of control point 11 as equations 5 and 6 did for control point 8.

Local Optimization

Local optimization on the other hand, calculates the value of system variables it is responsible for to ensure the best performance of its subsystem. Thus, in our example, we are trying to find the optimal power level at a given power source so that loads of direct responsibility are met and the subsystem cost is minimized. The local objective function for each subsystem, equation 9, is described below such that ***P*** is the total power generated at the power source in question and will fulfill its demand responsibilities.

$$\min \quad \alpha + \beta P + \gamma P^2 \qquad (9)$$

The constraints described below are in place to ensure local loads are serviced and power source capacity is maintained. Equations 10 and 11 ensure that demand at the local sites is met by the power source and are found in the local optimization programs for microsources 8 and 11, respectively.

$$P_{Utility} + P_{MS8} \geq Load_a + Load_c \tag{10}$$

$$P_{MS11} \geq Load_b + Load_d \tag{11}$$

Each describing the demand constraint for a different local subsystem, equations 10 and 11, are similar in practice. In equation 10, a given utility power parameter is considered. First, to understand why utility power is considered in the local program for microsource 8, one must consider what a similar program would return for the utility subsystem. The result of such a program would indicate that no power should be supplied from the utility because there are no loads associated with this subsystem and thus not economical to produce power. Global optimization however, suggests that power should be supplied by the utility to obtain better system results. Thus, utility power is considered and included in the subsystem that would directly benefit from the supply. Conversely, equation 11 does not have a similar input parameter for power flowing in from microsource 8. This is because the goal is minimum cost. Microsource 8 will not generate any additional power than that which is demanded from its local loads. A similar argument holds for flow control through the control points. This flow is not considered in the local optimization problem since it is economical for a microsource to produce only enough power to satisfy direct demand.

$$MinP \leq P \leq MaxP \tag{12}$$

Equation 12 describes the capacity constraints for the local microsource. A binary variable, however, is not needed in local optimization, as it was in global. This is because there is no other power source to satisfy the local loads other than those being considered.

Optimization Results

Given that we are ignoring startup and transmission costs, global optimization results in lower cost than local optimization. See Figure 28, which graphically relates costs. In local optimization, the source closest to a load satisfies the load. In global optimization, power is generated at the more-economical source until it can produce no more, and only then is power generated at the less-economical source. Grid-connected costs are less since power from the grid costs less than power from either microsource; it is only the point constraints (places in the microgrid where power levels cannot be exceeded) that permit the microsources to generate any power at all when grid-connected. In essence, the linear programming approach satisfies our intuition about how power ought to be allocated based on cost economics, i.e., in the absence of market forces.

We would expect similarly reasonable outcomes if transmission and startup costs were included. Other concerns of interest are equitable utilization, in that all generators should be run approximately equal percentages of the time, and maintenance scheduling, in that every generator needs to be shut down for periods of time at regular intervals for maintenance. A reasonably managed system would take at least these costs into account. The agents would use an extended version of the above model in making allocation decisions.

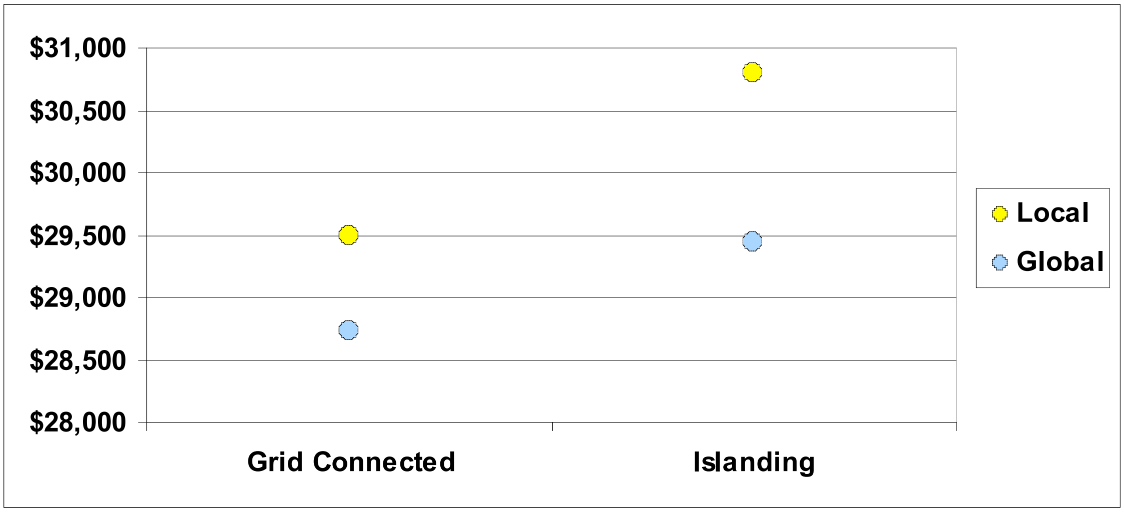


Figure 28: Cost of Five Hours of Satisfying the Example Loads

15.2.2. Maintaining Network State Information

Network state in the context of this project refers to the aggregate status information for all relevant elements of the power production network[20]. Each agent needs to maintain accurate state information about the part of the network for which it is responsible because its behavior rules and policies for taking action are based on state information.

To keep all agents informed of the network state during the demonstration, we would ideally use strong broadcast (e.g., Bracha's Broadcast protocol) to distribute information. In the absence of a guaranteed-outcome broadcast protocol, implementers would design a reliable communication mechanism and the agents would use multicast. The information of interest is the same as what's distributed by the "Raise Alarm" and "Report Trend" protocols (see Section 10 for protocol descriptions).

The demonstration illustrates the behavior of a single cooperative of cells made up of hardware controlled by agents. The network state contains information about hardware elements represented in the Matlab simulation[21] generator set points, generator output levels, load values, and line flow values, each associated with a particular hardware element.

Each agent involved in controlling part of the power network needs to maintain a data structure to contain state information about (a) network elements it directly controls and (b) other parts of the network that affect the parts it controls. For the demonstration, every agent maintains state information regarding the entire network[22].

---

[20] Of course the agents use an information network to communicate with one another and with the electric power network, and its status is also relevant to operation of the power grid. We defer the topic of maintenance of the information network.

[21] Network status at levels above the lowest will be represented in terms of the network's constituent cooperatives and cells and will not contain information about individual devices.

[22] As a network grows, it's neither necessary nor desireable that an agent maintain state information for the entire network. We expect agents will maintain *state* information about those parts of the network they directly control and *status* information, in the form of behavioral

In the demonstration, all agents get all information. This is not viable in general, but serves as a base case. Some distributed algorithms do not require that all participants have complete information, which mitigates the difficulty somewhat. See, for example, the distributed maximum network flow algorithm discussed in [Armbruster], where agents need to know only selected values for neighboring regions. Time stamps based on synchronization or a shared clock can be used to impose cutoffs so that all agents can agree upon "current" information vs. "imminent but possibly not fully distributed" information. Another possibility is to use an asynchronous consensus protocol, such as "Agreement on a Core Set", but this is expensive in terms of bandwidth and time. See [Cachin] for a discussion of broadcast protocols.

In general, an agent transmits state information when the state of something for which the agent has direct responsibility changes. Mere variation in an object's state value is not necessarily grounds for announcing a change; some variation may be allowed to account for sensor variability and minor power fluctuations. In addition, we expect to transmit state information on a scheduled basis if we have no noteworthy changes to excite a change-based trigger. We expect all such state information transmissions to be sent using Bracha's Broadcast so that every agent can predict what the others know.

Each agent requires at least[23] three network state data structures to accommodate the different stages of completion of its internal representation of the network.

- State data for which an agent is responsible but hasn't started a Bracha's Broadcast to inform the others.
- State data for which an agent is participating in a partially completed BB protocol.
- State data on which a BB protocol has been completed.

Depending on the rate of change of sensed information and the rate of information exchange mandated by policy, the agents could be in either more-or-less constant agreement or more-or-less constant disagreement. Experiments to determine what rates of change can be supported by a given agent implementation and policy are needed.

Open Issues: Group authority over breakers would be advantageous in a deployed system. Participants that are "behaving badly" by not reporting state information or not following established plans can be excluded by the others. This suggests that each player should maintain contingency plans so that when a participant is cut off from the co-op the remaining players can adopt a known configuration.

15.2.3. Achieving Consensus on Network State

Consensus is achieved by executing a broadcast protocol, such as Bracha's Broadcast (BB), of the network state to the parties among whom consensus is being established, each element of which has itself been established among these same parties by BB. The immediate question is: Whose network state gets broadcast?

A group signature can be performed on the network state to establish an "official" picture if an adequate number of participants are suitably informed prepared. This signed picture

---

expectations, about neighboring cells and co-ops that they connect to.

[23] Other similar network state data structures will be needed, e.g., to represent initial and final states for planning, to declare contingencies, and to report past states.

should be distributed either through a reliable network or using strong broadcast. A timely commonly-held network state image is important for future state prediction and planning, so it should be assigned high priority in terms of where strong broadcast should be applied.

If rates of change and broadcast schedules are benign, this point is all but moot. Most of the time, the agents will have consistent information; any of them can broadcast their accepted data at any time as the consensus candidate (using random election to choose the broadcaster) and the others will have the same values, because the data won't have changed from the last update. Any active BB protocols should be allowed to complete before the initial sending.

The potential for difficulty increases as rates of change increase, especially system-wide; the agents will be broadcasting often and the BB protocol takes some time to complete. While it is true that any datum for which BB has completed will eventually be accepted by all good players, there might be more recent information either just received from an agent's own sensors or for which the BB protocol hasn't finished. It can also easily be the case that the collections of accepted information held individually by the agents are not identical to one another, even if each agent is using only the most recent information for every device.

Our general point in raising this issue is to point out that the only way to avoid it is to use a protocol for communication state that is assured to complete faster than the rate of change of phenomena of interest. Otherwise, the agents must possess facilities for recognizing that they are in a circumstance where the rates of change of relevant phenomena is outstripping their ability to keep up. In addition, the agents must have policy for determining what they should do to accommodate this sort of circumstance. For example, they might stop using BB, they might take action without waiting for BB to complete, or they might base their actions on individual elements of state data for which BB is complete but for which a complete network state object has not yet been assembled. Each course of action would engender different recognition capacities, produce different outcomes, and suffer different kinds of failure.

Open Issues: If it becomes impossible to reach consensus on state due to precision problems, information propagation problems (delays or faulty links), or adversarial activity, then approximate consensus, dynamic collaboration, and negotiation protocols can be used. This would require research to discover (or develop) appropriate distributed algorithms.

15.2.4. Planning

There are three operation time scales of interest in the electric power domain:

Near-Instantaneous (less than a second) – This is the time frame in which the system has to handle minor or routine load fluctuations, presumably by having generators preconfigured to respond in a certain way to local load changes. In emergency scenarios this is the approximate time frame in which breakers would need to be thrown; for example, to cut off a non-critical load in order to mitigate the problem of a local generator failure. The agents are not expected to respond in this time frame but can observe its effects.

Short-term (seconds, up to a minute) – This is the time frame in which conditions that require no more than reporting and invocation of existing group plans can be addressed. In practice, an event is observed and reported to tactical peers, and the agents to whom the event has been reported take predetermined steps. This is the notional process for responding quickly when a generator fails, given that a contingency plan for that scenario already has been distributed and the failure event alone will trigger a designated response.

Long-term (minutes to years) – This is the time frame in which plan generation and distribution can occur, enabling changes to the power system to be handled according to policy, assuming the system has been stabilized in the near-instantaneous or short-term time frames. Contingency plans should already be in place for specific emergencies, and these might buy time for the planning process. Replanning, agreeing upon the final plan, distributing this, and putting the plan into action is expected to take several seconds, if not minutes. In the demonstration this process takes only a few seconds, but this is not a hardware-in-the-loop situation.

For example, when a generator fails, the immediate response should be to cut the breaker to the generator and the local low-priority load; the short-term response should be to report the problem and have everyone enact contingency plans (either cutting more low-priority loads or not, and raising the maximum limits on generators); and the long-term response should be to come to consensus and enact a plan that meets the distribution constraints, everyone's critical load, and as many non-critical loads as possible, with the non-generating tent's non-critical load being preferentially unsupplied.

The planning process is roughly cut into several pieces:

A. Achieving and maintaining a consistent system state model (see Section 15.2.2)

B. The approximate planning algorithm:

1. Compute generator set points based on expected loads and available sources
2. Compute load flow and check against distribution constraints
3. Iterate this until distribution constraints are met, adding constraints on maximum transmission and removing problem loads from one iteration to the next
4. Add tolerances to plan based on slack buses, known features of the environment, etc.

One way of handling failure to generate a plan is to cut someone out of the picture entirely, starting with individuals that are causing difficulty, and resume the planning process.

Open Issues: Many issues with power policies and plans, decentralized planning, planning given more complicated power systems, contract negotiation, etc. The research issues here are beyond the scope of this project.

C. Coming to consensus on the plan and distributing it to the actors

Given a deterministic planning process, sign the plan and use strong broadcast to distribute it.

Generating contingency plans involves executing the planning process against failure scenarios derived from the current system state. Each plan that results should be marked as hypothetical and annotated with a description of the failure that engendered its creation.

D. Plan execution and monitoring

Plans should be time-stamped to take effect some time after the agreed-upon cutoff for official state information. The tactical monitor/controllers should achieve the planned configurations—opening breakers, setting set points, etc.—as close to that time as possible.

Open Issues: Clock synchronization was not implemented in the demonstration, but this topic has been thoroughly researched. Large numbers of systems can be accurately synchronized using various approaches, even for situations that include bad players. Selection of a robust synchronization technique will require some research but presents no conceptual difficulties.

Of greater concern is cross-monitoring and constraining the effects of bad participants by merit of plan structure. Generating plans upon which misinformation or incomplete information has only a bounded effect is also an issue.

15.2.5. Monitoring

In our demonstration, a Matlab model is used to simulate the network environment. Agents use a planning algorithm distinct from Matlab to determine set points and predict load flow. The agent compares planned set points and load flows that it has computed with those it "perceives" from the Matlab simulation, which is acting as the "real world" environment. Agents' communication with Matlab must result in them perceiving suitable network state to compare with their expectations; when the environment no longer reflects the current operational plan, a new plan is generated.

The plant representation (ideal power sources with finite settling times, short transmission lines, ideal transformers, ideal switches/breakers, real and reactive loads) and microgrid use-cases should determine the plant modeling needs: powerflow analysis, fault analysis, and transient simulation. Transient modeling of the plant is non-trivial and there is still much to contemplate, study, and experiment with.

15.3. Agent Extensions

This section discusses the dimensions of extensibility required to create agents of each of the classes discussed in Section 12.

15.3.1. Agent Face Extensions

Each instantiated agent needs to be able to act in a number of roles. The roles are cross-matrixed with the organization types (cell, glob, and co-op; see Section 11) to result in the faces needed to support the appropriate organizational structure.

See Sections 11 and 12 for the subset of all possible combinations that are needed to execute the design.

15.3.2. Agents, Faces, Roles, Identities, Authority, etc.

Each agent needs data elements that represent the other agents with whom it is dealing. The agents use these models of the other agents to contain their knowledge about the capabilities of the other agents, for example, to whom information should go or whether a request comes from an authorized individual. The agent models would also contain the data each agent has about the states of the other agents.

Agent faces need to be instantiated as components of the models of other agents that each agent keeps. The initialization of these objects along with certificates, etc. can be set up when the functionality for the face is loaded, or it can be that certain pieces of functionality will be activated only when the proper face is instantiated (this would be advantageous, actually, because all the software could be loaded into all the agents, and selectively enabled to create a collective).

Agents with multiple responsibilities in the collective will be implemented with the combination of the functionality for the individual responsibilities, and may sometimes go through the motions of talking between agents when in fact it is both agents. Shortcutting this is an optimization step that can be left for later, but this strategy makes delegating roles and responsibilities easier. It does mean that messages being sent and received should have not only target agent information in the message but also some indication of the target face.

15.4. Goals and Planning

Operations and motivations will periodically create new goals for an agent's agenda. The representation and types of these goals need to be declared, and planning algorithms need to be written for each type of goal that may be formed. The planning algorithms need to always produce a list of grounded plan elements, and in particular this means that each planner needs a translation process that will map plan elements into operational schemata.

15.5. Operations and Protocols (Schemata, i.e. state diagrams)

The state diagram processing system is the most flexible and reliable way we have at the moment to implement conversation specifications (the Protocols of the Design document), and there will most likely be additional stateful operations that can be easily implemented in this way, such as setting and then confirming some control on a device.

15.6. Expectations

Most of an agent's expectations will be posted and removed by the operations an agent is executing. However, there will be a number of normative expectations that agents use to jump start protocols or detect standard anomalies, and those will need to be called out here, separately from the operations and protocols.

15.7. Ontologies

Foreign ontologies and the local ontology are implemented as package structures that capture all the class definitions of messages that can be sent from one agent to another, or received through sensation. To specify these ontologies, we declare all the class definitions and any special processes that must be used to reconcile objects (viewed as instances of foreign ontologies) with the local ontology.

In the initial implementation, we would expect the agents to be ontologically homogenous, and thus able to use packages with the same definitions and ontologies: all system agents would be "born" with identical knowledge. Later, as agents acting on behalf of different businesses and governments began to interact, some means to assure intercommunication would be needed. See the introductory paragraphs of Section 4 for discussion of closed vs. open system concerns.

15.8. Perception and Articulation

The abilities to respond to http requests and respond with html, and to send and receive dclos messages with other agents, should be provided by default in the framework. However, the additional need to interact with a simulated power system as if with physical devices means that the Matlab formats for the data sources of interest (and their associated control functions) will need to be implemented as framework extensions.

All interactions of Cell Representative agents, Cell Logistics agents, Co-op Legislator agents, and Co-op Operator agents occur through http, html, and dclos. No additional perception or articulation is required.

The Cell Operator needs to be able to interact via TCP/IP with Matlab to maintain a perceivable and manipulable virtual device. This is in addition to the default communication mechanism. Within the agent, percepts will be generated pertaining to virtual devices, and these devices will be acted upon. Below that abstraction barrier, Matlab TCP/IP transmissions must be parsed in order to identify changes to the device that must be recorded and perceived, and device actions need to be turned into Matlab-readable commands.

Perception related to a device needs to be shared with the Protection System, but actions on devices made by the Cell Operator should be subordinate to actions taken by the Protection System.

In addition to the default communication mechanism, the Protection System needs to be able to interact via TCP/IP with Matlab in the same way as the Cell Operator (see above for details). Actions taken on virtual devices by the Protection System should override those of the Cell Operator.

16. Implementing the Demonstration

Scenario: An authorized human wishes to take a generator offline and communicates this to the agents. The agents reconfigure the system to accommodate the loss of capacity and shut down the generator.

The demo consists of four agents, all talking to a power simulation, each in charge of a separate portion of the network that includes a source and two loads, sharing status information and coordinating their actions.

16.1. Demonstration Implementation Discussion

In the course of this project, we implemented a small demonstration system. This demonstration serves as a proof of principle—agents are interacting with a human user and a simulated microgrid—and to show that a complete implementation of the system design that resulted from Gaia process could be accomplished. The demonstration system lacks all the features we had hoped to include, but the reasons for this are engineering reasons, not conceptual problems with the framework.

Externally to the project it was decided that money needed to be pooled between several agent projects (including ABCDIR) to advance the state of AAF, a new experimental version of our agent framework. It was also decided externally to our project that AAF would be used for ongoing agent development on all projects in our lab, including ABCDIR. This shaped the demonstration and schedule.

For example, we had hoped to use the Bracha's Broadcast procedure to demonstrate strong broadcast, which we know will avoid some byzantine failures. We did not, however, include this in the demo because the newly delivered framework could not readily accommodate the existing Bracha's Broadcast protocol implementation we had previously developed. We also did not establish a Domain Authority Group for the same reason.

Nevertheless, a discussion of the implementation should serve to illustrate what we had envisioned, some of the challenges in taking an agent system from the drawing board to a prototype, our particular design for the demonstration and some of the details of its implementation, and the lessons learned in the process.

16.2. Demonstration Objective

The design reflected in this document describes a large software system that was understood at the outset to be beyond the scope of this LDRD to implement in its entirety. There are open research questions that need to be answered before this design can be achieved, such as the appropriate use of multi-party cryptography and other protocols in the context of a security policy for the management of electric power, or how local generation and transmission constraints can be composed to form regional constraints, and how a fixed policy at the regional level can be used to set policies at the local power system level.

In choosing the use case for the demonstration, we agreed that a particularly interesting aspect of the multi-agent system was the process by which agents shared information to arrive at a common picture of a cooperatively managed power system. A simulation of an electric power system would be created and multiple agents would interact with the simulation and one another in such a way that the agents all had a limited view of the power network. Each agent would need to consider what actions to take based on the composite picture it had arrived at given the other agents' descriptions of state.

16.3. Detailed Demonstration Script

1. A User communicates with a System POC agent via a web browser and goes through the *Start User Interface* protocol. The System POC should have information about the current configuration and state of the network and present this to the user as a web site.
   - 1.1. The User opens web browser and points at http://knownmachineforco-opcellrep/co-opPOCrootpath (a totally fabricated URL root for the "System POC" role of the "Cell Rep" for the co-op). This results in an http "GET" string being sent to the appropriate agent[24], designated "Agent-1" in this script.
   - 1.2. Agent-1 receives and responds to the http request.
     - 1.2.1. Agent-1 accepts TCP connection on port 80,
     - 1.2.2. recognizes data as http,
     - 1.2.3. delimits http request from stream and forms percept,
     - 1.2.4. parses http request from percept,
     - 1.2.5. matches object percept/material percept against standing (normative) expectation that it will be sent http requests, and
     - 1.2.6. creates http service schema and re-submits percept to expectation matching queue.
     - 1.2.7. Agent-1 provides http service
     - 1.2.8. Agent-1 marshals and formats data into html bolus
       - 1.2.8.1. Data comes from the network state information each agent is maintaining [see Section 15.2.2, "Maintaining Network State Information," p. 90). For the demo, this will be the generator output levels and setpoints, power flows, whether the controllable breakers and switches are open or closed, and the loads.
       - 1.2.8.2. Initially, send available data as text and add a URL "Shut down generator 1" will satisfy the needs of the demonstration. We assume this will become more elaborate as time and funding allow.
     - 1.2.9. Agent-1 sends html bolus to the user through connection established in 1.2.1.
     - 1.2.10. Using the schema mechanism, agent-1 posts an expectation of receiving http request over this TCP connection[25]
2. The User initiates the *Submit User Request* protocol, the request being to and shut down a generator. In the demonstration the user initiates this protocol by selecting a "shut down generator 1" action on the interaction web page. The System POC uses Bracha's Broadcast (BB) to pass the request on to the agents collectively filling the role of Strategic Planner (SP)[26].

---

[24] Although the "cell representative" for the co-op could be a distributed function, for the demo we designated a single agent to act as cell rep when the agents were initialized.

[25] This implies location and type specificity, which gives this interaction between the user and agent-1 the properties of a session

[26] Several agents—specifically, the four that are operating tent cells—participate in the Strategic Planner (SP) Domain Authority Group (DAG).

2.1. User points at http://known_machine_for_co-op_cell_rep/shut_down_generator_1. This results in an http request being sent to Agent-1 over the connection established in 1.2.1 above.
2.2. Agent-1 forms a percept from the http request and matches it to the expectation created in 1.2.10, which reactivates the schema associated with the session.
2.3. Agent-1 runs the activated schema, the next step of which is to activate a second schema that uses Bracha's Broadcast (BB) to inform the SP that generator-1 is to be shut down. For the demo, BB completes successfully which results in the SP agents all receiving and acting upon an authorized request to shut down generator 1.
2.4. The SP agents achieve consensus on desired network state, which is produced by merging the state information implied by extrapolating the effects of the user request[27] with the state predicted by extrapolating the current plan (i.e., the plan as it was before the user request). For this use case in its simplest form, the effects of the user request are based on the knowledge that a producing generator will not be producing if the request is executed.
2.5. The SP agents request the current network state from the SM. Arriving at and maintaining an accurate, commonly held picture of the network state is non-trivial. The manner in which this is done may affect the current process. See Section 15.2.3, Achieving Consensus on Network State, p. 91.
2.6. The Strategic Monitor (SM) informs the SP of the current network state.
    2.6.1. Each agent in the strategic monitor role prepares a message that includes the information it has accepted as part of a strong broadcast procedure about the network as well as any information it has direct oversight of as a TMC that it has not yet had a chance to submit to the SM group.
    2.6.2. Each SM agent uses strong broadcast to send this message to the SP group.
    2.6.3. The SP agents achieve consensus on a model of the network that represents the network in its current state and a planning goal[28] (see "Achieving Consensus on Network State").
    2.6.4. After a certain elapsed time (duration unknown, but the idea is to give BB a chance to complete for everyone), the SP group completes an agreement on a core set (ACS) protocol on the set of messages being sent by the SM group.
    2.6.5. Upon completion of the ACS protocol, the core set of model/request messages are combined deterministically to create the official model/request value. This consensus result must be sufficient for planning

3. When a consensus has been reached, the planning algorithm is executed (see "Planning", page 92).
    3.1. Compute generator set points based on expected loads and available sources
    3.2. Compute load flow and check against distribution constraints
    3.3. Iterate this until distribution constraints are met, adding constraints on maximum transmission and removing problem loads from one iteration to the next
    3.4. Add estimated tolerances to plan based on slack buses, etc.
    3.5. Add flow constraints to the set point computation's input to resolve problems between iterations.

[27] Such a merge can result in inconsistencies, e.g., when user requests result in policy violations.
[28] For the demo the SP agents are the same as the cell operator agents reading the sensors.

3.6. SPs achieve consensus on the plan. Various failures to achieve consensus are ancillary use cases.

4. The *Distribute Plan* protocol is launched when a plan consensus is reached.
   4.1. The SPs sign the plan using group signature.
   4.2. The SPs use strong broadcast to send the signed plan to the TMCs, SMs, etc.[29]

5. Execution and monitoring of the new plan commences.
   5.1. The TMCs make the control changes for generators and breakers and report status as things progress.
   5.2. Generator is shut down for maintenance.
   5.3. The SMs observe the network has reached the intended goal state and collectively sign a command to the cell operator of the generator to shut the generator off.
   5.4. The SMs all transmit (no multicast or anything) the signed message to the CO, who does so and reports that fact using BB.

6. The SMs record the new network state, and report this back to the SPOC.

7. The SPOC report the results to the User.

16.4. Demonstration Implementation Stages

16.4.1. Pre-demo Integration

As we worked with Sandia's Active Agent Framework (AAF) core team to document the requirements our intended demonstration would impose on the agent system, both in terms of framework functionality and the necessary extensions we would use, we began work on the interface between the AC power simulation, our lisp environment, and the web browser we would use as a demonstration interface. This "pre-demo" was used to implement and test the end-to-end data flow in the system before adding this functionality to an agent.

Although the Matlab packages used to model AC power flow did have the capacity to generate the common data format (CDF), the available tcp/ip package for Matlab was unable to serve multiple connections. We wrote a service wrapper in perl to handle multiple independent connections and serialize requests for Matlab.

- Matlab talking over the network via a perl service wrapper
- Matlab responding to lisp with a CDF stream
- Lisp reading a CDF stream and constructing a network data structure
- Lisp initiating changes in the AC model used by Matlab
- Lisp dumping a network as an HTML table in a web page
- Identify mutually-compatible versions of everything

[29] The TMCs and SMs get the entire plan. It's not up to the SPs to remember which agents have which parts of the network. This implies that any agent serving a role that is supposed to act on a plan needs to know how to determine what parts of the plan are its responsibility.

16.4.2. Demonstration Stage 1

The first demonstration incorporated the process of the pre-demo into a simple agent. The benefit of agency in this case is not significant, but this was a logical stepping-stone on the path towards a multi-agent demonstration: it allowed us to test and tag the appropriate version of the AAF and familiarize ourselves with its use.

The particulars of how the agent interacted with Matlab and the user's web browser are very similar to the pre-demo, but with additional layers of abstraction dividing the pre-demo into different modules, or 'sites.' The majority of this did not change for the second demonstration.

- Bi-directional communication between Matlab and Lisp
- Bi-directional communication between Lisp and AAF sites with browser interfaces

16.4.3. Demonstration Stage 2

The final demonstration incorporated an existing agent-to-agent communication site to demo 1, so that two agents could communicate with one another and added one agent. The code from demo 1 was altered so the two agents had jurisdiction over separate portions of the power network. Each agent informed the other of the observed status of its portion of the network so either agent could produce content for the user. When the user submits a command (in this case, to shut down a generator), the command is shared between the agents and the agent that is the tactical controller and monitor for the appropriate resource acts on the command.

- Separate CDF data reading from the selection of "observable" portions of the network
- Add functionality so agents exchange observable network section data with one another
- Each agent combines shared network sections into its own model of the power system
- One agent interacts with a user via web browser to receive the shutdown command and share it with the other agent
- The agent who presides over the affected network section acts on the command

16.5. Demo visualization

We invite the reader to compare Figure 3, Figure 4, Figure 5, Figure 29, and Figure 30: They are all representations of the four-tent power system.

During the final summer of the project, students working on a related project involving agents and microgrids (see [Miller] and [Phillips]) presented a visualization mechanism they had developed. We suggested they display the four-tent scenario before and after the generator shutdown effected during the demo. The results are shown in Figure 29. Note the absence of the left-most triangle of the upper array compared to the bottom array; this represents the generator that's been shut down. This display mechanism accepts files in the Common Data Format (CDF), which is IEEE's standard format for published standard networks (such as the IEEE 118-bus network shown in Figure 1).

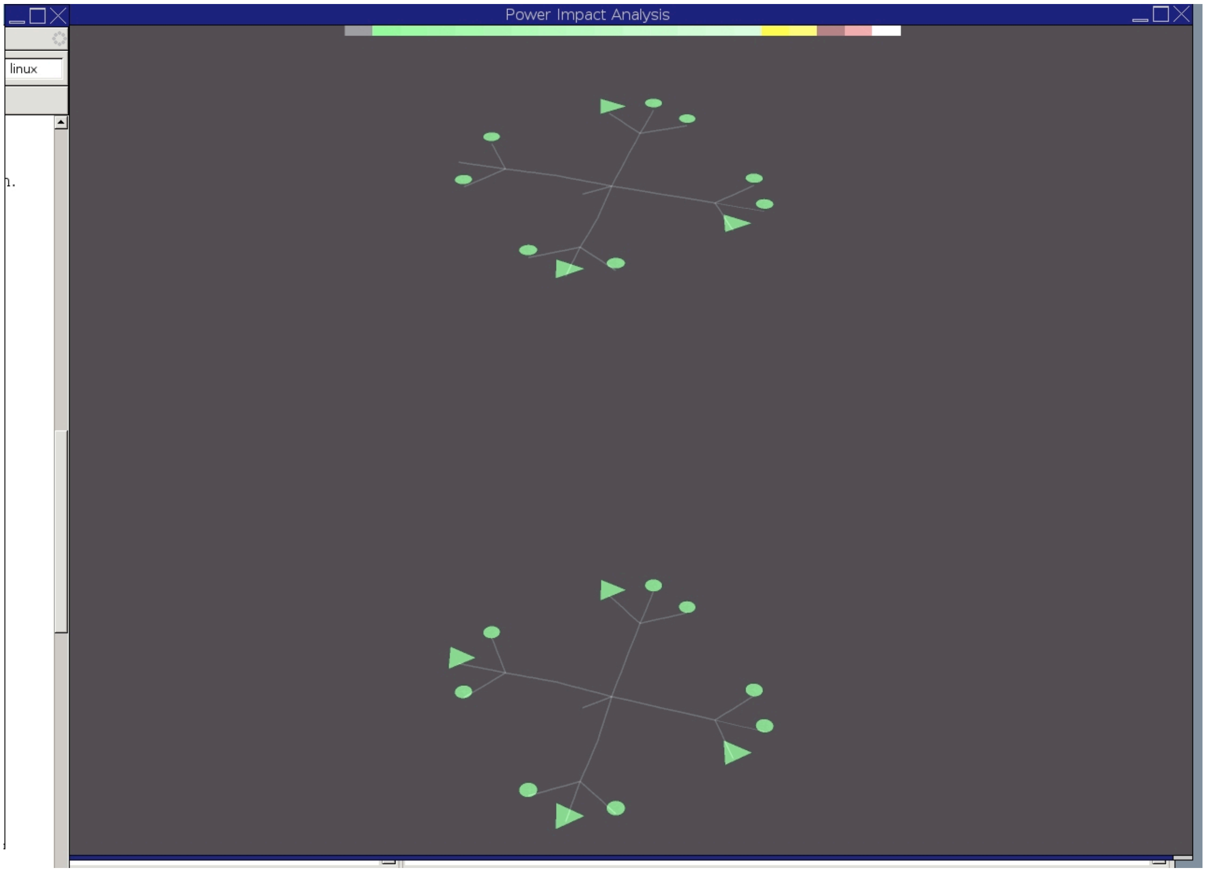


Figure 29: Four-tent network before (bottom) and after (top) generator shutdown

Bus #	Type	Load MW	Load Mvar	Gen MW	Gen Mvar	Base kV
1	0	0	0	0	0	2.400
2	2	0	0	0.03	0	0.480
3	0	0	0	0	0	0.480
4	1	0.008	0.006	0	0	0.120
5	1	0.008	0.006	0	0	0.480
6	0	0	0	0	0	2.400
7	2	0	0	0.03	0	0.480
8	0	0	0	0	0	0.480
9	1	0.008	0.006	0	0	0.120
10	1	0.060	0.006	0	0	0.480
11	0	0	0	0	0	2.400
12	2	0	0	0.03	0	0.480
13	0	0	0	0	0	0.480
14	1	0.008	0.006	0	0	0.120
15	1	0.008	0.006	60	0	0.480
16	0	0	0	0	0	2.400
17	2	0	0	0.03	0	0.480
18	0	0	0	0	0	0.480
19	1	0.008	0.006	0	0	0.120
20	1	0.008	0.006	0	0	0.480
21	0	0	0	0	0	2.400

Table 3: CDF of Four-tent network *before* generator 2 shutoff

Bus #	Type	Load MW	Load Mvar	Gen MW	Gen Mvar	Base kV
1	0	0	0	0	0	2.400
2	2	0	0	0	0	0.480
3	0	0	0	0	0	0.480
4	1	0.008	0.006	0	0	0.120
5	1	0.008	0.006	0	0	0.480
6	0	0	0	0	0	2.400
7	2	0	0	0.05	0	0.480
8	0	0	0	0	0	0.480
9	1	0.008	0.006	0	0	0.120
10	1	0.060	0.006	0	0	0.480
11	0	0	0	0	0	2.400
12	2	0	0	0.03	0	0.480
13	0	0	0	0	0	0.480
14	1	0.008	0.006	0	0	0.120
15	1	0.008	0.006	60	0	0.480
16	0	0	0	0	0	2.400
17	2	0	0	0.03	0	0.480
18	0	0	0	0	0	0.480
19	1	0.008	0.006	0	0	0.120
20	1	0.008	0.006	0	0	0.480
21	0	0	0	0	0	2.400

Table 4: CDF of Four-tent network *after* generator 2 shutoff

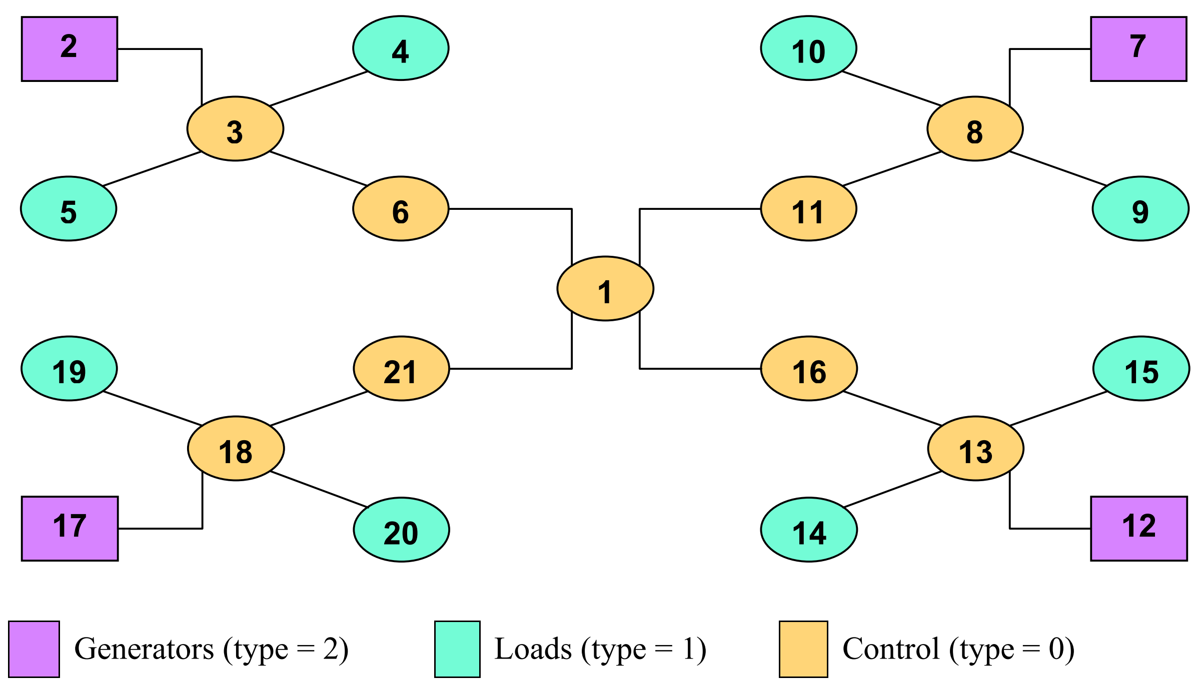


Figure 30: Four-tent schematic for table interpretation

16.6. Demo execution

The user image of the operational demo is rather anticlimactic: A few numbers change on a table. A user approaches a standard browser (e.g., Explorer or Firefox), accesses a specific URL, and receives a tabular layout of a Common Data Format (CDF) file similar to Table 3. Below the table is a single button labeled “Shut down generator”. When the user clicks the button, the table is updated to a display similar to Table 4. The user may wish to examine the tables for differences; in fact the only change is the “Gen MW” column for nodes 2 and 7, which are the generators of interest (see Figure 30). The generator at node 7 is in load-following mode while those at 2, 12, and 17 are in fixed-point mode. Totals for power produced and power consumed do not quite match because of round-off error in the displayed values.

What has happened is that one of the agents has received the user’s signal to shut down the generator, which is pre-designated in the demo. Each of the agents, who have been sharing information since before the user contact, develops a single-step plan to achieve the user’s goal of having the generator shut down, i.e., “Shut down the generator.” The agents determine that satisfying the existing loads after the generator is shut down is within the capabilities of the remaining currently running generators. The agents share these results with one another and discover that they agree, at which point the agent in charge of the generator of interest shuts it down.

— This page intentionally left blank —

17. Conclusions and lessons learned

Although much work remains before coalitions of autonomous agents should be allowed to manage real-world power systems, we believe the principles, framework, and framework extensions described in this report lay effective groundwork for this task.

This report contains the specification of an agent-based infrastructure management system. Its contents permit an optimistic outlook towards construction of an agent-based infrastructure management system, but a good deal of work remains before such a system can be deployed. See Section 18 for a discussion of some of the challenges. The authors have constructed and demonstrated an agent coalition capable of performing some of the tasks necessary for operating an electric power microgrid.

Control and *management* are two distinct sets of functionality, and both are necessary to operate distributed infrastructures. Many control tasks can be performed by wholly automated mechanisms based on local conditions, but management requires knowledgeable oversight by autonomous, situationally aware entities that can communication with one another and act quickly and with assurance. This is essentially the definition of a distributed agent-based system.

A distributed coalition of agents can perform in a safe, secure, timely manner many of the tasks necessary to manage and operate distributed infrastructures, and an agent-based infrastructure management system would enable timely fault recovery and, in some cases, prevention of faults and cascades. This report gives the design of an agent-based system capable of these tasks.

The agent-based infrastructure management system design produced by this project addresses and to some extent resolves the following issues of agent functionality for distributed infrastructures in general and for microgrids in particular:

- the function of power-system management agents in steady state and under contingency;
- the nature of their interactions with one another and with humans;
- the essential organizing principles for groups of infrastructure-managing agents;
- the means by which the agent coalition provides security; and
- the means by which agents enforce both operations policy and security policy.

— This page intentionally left blank —

18. Future work

Much work remains to construct trustworthy, deployable agent systems with the breadth, depth, and fault tolerance to manage distributed infrastructures. This statement stands despite the contributions of many towards designing and building relevant agents.

A great deal of this work involves realizing the framework outlined in this report:

- Construct a set of use cases that captures the full functional requirements of an agent-based power management system;
- Build and demonstrate a system capable of executing the extended set of use cases;
- Select from the literature, or otherwise discover, the best algorithms for system functions, particularly fault isolation and recovery, and integrate these into our framework;
- Develop and exercise a full set of liveness and safety conditions;
- Mitigate simulated contingencies that closely replicate actual known events,
- Evaluate the benefits and risks of using an agent-based management system; and
- Demonstrate agent-based management of actual hardware-based power systems.

Storage is an important element of any electric power system, and is more important for smaller systems. Storage capacity enables the system to respond quickly to sudden load increases and generation shortfalls and to provide stable power. This needs to be included in the environment and the ramifications carried throughout the agent system.

We have all but ignored *commerce*: The formation and execution of power transactions. Significant space has been devoted to this topic in the literature. We would start with the framework presented here and incorporate the ability to participate in auctions, to make and fulfill contracts, etc.; in other words, to participate in a market. This raises the prospect of a cell/co-op-based power economy. This is a fundamental need for future consideration.

An important element that we have touched on in this report—although barely—is *scaling*: Laboratory proof-of-principle experiments involving a few dozen agents shed little light on how several thousand agents manage a large distributed power grid. The cell/co-op concept described in this report resolves this, but only conceptually: the agents should be organized as cells, which are organized as co-ops, which can themselves form co-ops. Notwithstanding this report and the multiparty behavior we have demonstrated, we need algorithms and protocols that enable all the necessary cooperative behavior for a large, distributed system.

Perhaps the most pressing need in the scaling area is for *utility functions* to determine the value of relevant options or choices. Utility functions are needed for *deliberation*, the weighing of alternatives for their projected utility. They should involve both the tangible costs of producing and transporting power and the costs that are difficult to quantify, such as risk (risky actions are more costly), opportunity cost, convenience, legal issues, and so forth. It is not difficult to invent reasonable utility functions for a particular agent or group.

Individuals wishing to cooperate need *group utility functions*. One approach is to combine the individual utility functions of the member entities, but this is not always mathematically possible. Combining linear utility functions appears to be possible if there is a defined global goal, though non-trivial if the agents make discrete choices. Pareto-optimal co-evolution may enable robust combined non-linear functions. In any case, future work must seek a theory of group utility function to allow groups of entities to make decisions.

— This page intentionally left blank —

19. References

[Amin99] Amin, M.; "National Infrastructures as Complex Interactive Networks"; *Automation, Control, and Complexity: New Developments and Directions*; Samad and Weyrauch (Eds.), John wiley and Sons, 1999.

[Amin01] Amin, M.; "Toward Self-Healing Energy Infrastructure Systems"; *IEEE Computer Applications in Power*, January 2001.

[Amin02] Amin, M.; "Security Challenges for the Electricity Infrastructure"; Special issue of *IEEE Computer Magazine on Security and Privacy*, April 2002.

[Armbruster] Armbruster, A.; Gosnell, M.; McMillin,, B.; and Crow, M.; "The Maximum Flow Algorithm Applied to the Placement and Steady State Control of FACTS Devices"; *Proc. 2005 North American Power Symposium*; October 2005. http://web.umr.edu/~ff/Power/Papers/Tpower04.pdf

[Arnheiter] Arnheiter, T.; "Modeling and simulation of an agent-based decentralized two-commodity power market"; *Proc. Fourth International Conference on MultiAgent Systems*; 2000.

[Brazier] Brazier, F.; Cornelissen, F.; Gustavsson, R., et al.; "Agents negotiating for load balancing of electricity use"; *Proc. 18th International Conference on Distributed Computing Systems*; 1998.

[Bretas] Bretas, A.; *Robust Electric Power Infrastructures. Response and Recovery during Catastrophic Failures*; PhD dissertation, Virginia Polytechnic Institute, 2001.

[Butler-Purry] Butler-Purry, K.; Sarma, N.; and Hicks, I.; "Service restoration in naval shipboard power systems"; *IEE Proc. Generation, Transmission, and Distribution, Vol. 151*, No. 1; 2004

[Cachin] Cachin, C; Kursawe, K.; Petzold, F.; and Shoup, V.; "Secure and Efficient Asynchronous Broadcast Protocols"; *Proc. 21st Annual International Cryptology Conference, Advances in Cryptology, CRYPTO 2001*; 2001.

[Chaloupek] Chaloupek, J.; Tauritz, D.; McMillin, B.; and Crow, M.; "Evolutionary Optimization of Flexible A/C Transmission System Device Placement for Increasing Power Grid Reliability"; Submitted to the *2005 International Workshop on Frontiers in Evolutionary Algorithms*; July 2005. http://web.umr.edu/~ff/Power/Papers/FEA2005.pdf

[CSB] U.S. Chemical Safety and Hazard Investigation website; Completed Investigations page. http://www.csb.gov/index.cfm?folder=completed_investigations&page=index

[deAzevedo] de Azevedo, G.P.; Feijo, B. and Costa, M.; "An agent-based approach to EMS in open environments"; *Proc. International Conference on Electric Power Engineering;* Budapest; 1999.

[Dimeas04] Dimeas, A. and Hatziargyriou, N.; "A Multiagent System for Microgrids"; *IEEE Power Engineering Society General Meeting*; 2004.

[Dimeas05] Dimeas, A.L. and Hatziargyriou, N.D.; "Operation of a Multiagent System for Microgrid Control"; *IEEE Transactions on Power Systems;* 20(3): 1447-1455; 2005.

[DOE1] Transmission Reliability Multi-Year Program; U.S. Department of Energy; May 2001. www.electricity.doe.gov/documents/tr_myp.pdf

[DOE2] *Electricity Delivery and Energy Reliability Research, Development and Analysis*, U.S. Department of Energy Funding opportunity DE-PS02-05CH11270; June 2005. https://e-center.doe.gov/iips/faopor.nsf/0/83189341197E7F7D85257014006FD7D6?OpenDocument

[EIA04] *Existing Electric Generating Units in the United States 2003*; Energy Information Adminiatration website, 2005. http://www.eia.doe.gov/cneaf/electricity/page/capacity/capacity.html

[ENVIR] "Capstone Microturbine: Opening Up Cogeneration Opportunities – Product Review"; *Environmental Building News*; Oct 2000 http://www.buildinggreen.com/auth/article.cfm?fileName=091008a.xml

[Hotta] Hotta, K.; Nomura, H.; Takemoto, H.; Suzuki, K.; Nakamura, S.; and Fukui, S.; "Implementation of a Real-time Expert System for a Restoration Guide in a Dispatching Center"; *IEEE Transactions on Power Systems, 5(3)*; Aug 1990.

[Kraus] Kraus, P.; et al.; *Analysis of Electric Machinery*; ISBN 0780311019; IEEE Press, 1995.

[Kueck] Kueck, J.D. et al.; *Microgrid Energy Management System*; ORNL/TM-2002/242; January 2003. http://www.ornl.gov/sci/btc/apps/%20Restructuring/ORNLTM2002242rev.pdf

[Lasseter02] Lasseter, R.; Akhil, A.; et al.; *White Paper on Integration of Distributed Energy Resources: The CERTS MicroGrid Concept*; prepared for the Transmission Relibility Program, U.S. Department of Energy; April 2002.

[Lasseter04] Lasseter, R. H.; and Piagi, P.; "Microgrid: A Conceptual Solution"; in *Proc. Power Electronics Specialists Conference*; June 2004.

[McMillin] McMillin, B., and Crow, M.; Fault Tolerance and Security for Power Transmission System Configuration with FACTS Devices; *Proc. 2000 North American Power Symposium*, pp. 5.1-5.9; October 2000. http://web.umr.edu/~ff/Power/Papers/ETIpaper.pdf

[McDermott] McDermott, T.; *A Heuristic Nonlinear Constructive Method for Electric Power Distribution System Reconfiguration*; PhD Dissertation, Virginia Polytechnic Institute, 1998.

[Miller] Miller, S.; Service, T.; Atkins, W.; *AnVebidas: a Generic Agent Framework for the Analysis and Visualization of Emergent Behavior*; Sandia National Laboratories Center for Cyber Defenders project report; June 2005.

[NERC] *Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations*; U.S.-Canada Power System Outage Task Force, April 2004. http://www.nerc.com/~filez/blackout.html

[NTSB1] *National Transportation Safety Board Pipeline Accident Report PAR-03/01*, February 2003. http://www.ntsb.gov/publictn/2003/PAR0301.pdf

[NTSB2] *National Transportation Safety Board Safety Recommendation P-02-04 and -05*; Engleman, Ellen; October 2002. http://www.ntsb.gov/Recs/letters/2002/P02_04_05.pdf

[Ong] Ong, C.; *Dynamic Simulation of Electric Machinery using Matlab/Simulink*; ISBN 0137237855; Prentice Hall, 1997.

[Petrov] Petrov, V.; Richter, C.W. and Sheble, G.B.; "Predatory gaming strategies for electric power markets"; *Proc. International Conference on Electric Utility Deregulation and Restructuring and Power Technologies*; 2000.

[Phillips] Phillips, L.; Baca, M.; Hills, J.; Margulies, J.; Tejani, B., Richardson, B.; and Weiland, L.; *Analysis of Operations and Cyber Security Policies for a System of Cooperating Flexible Alternating Current Transmission System (FACTS) Devices*; Sandia National Laboratories Technical Report SAND2005-7301; published December 2005.

[Rehtanz] Rehtanz, C.; *Autonomous Systems and Intelligent Agents in Power System control and Operation*; Springer-Verlag, 2003.

[Rotkowitz] Rotkowitz, M.; and Lall, S.; "Decentralized Control Information Structures Preserved Under Feedback"; *Proc. IEEE Conference on Decision and Control*; 2002.

[Rudnick] Rudnick, H; Harnisch, I.; and Sanhueza, R.; "Reconfiguration of Electric Distribution Systems"; *Revista Facultad de Ingenieria, U.T.A.* (Chile); Vol. 4; 1997

[Ryan] Ryan, M.; Marksoe, S.; Cheng, Y.; Liu, F.; and McMillin, B.; "Structured Object-oriented Co-analysis/Co-design of Hardware/Software for the FACTS Power System"; *Proc. 29th Annual Computers, Software, and Applications Conference*, July 2005. http://web.umr.edu/~ff/Power/Papers/SandiaModelReport.doc

[Saadat] Saadat, H.; *Power Systems Analysis, 2nd Ed*; ISBN 0072848693; McGraw-Hill 2002.

[Safire] Safire, William; *The Farewell Dossier*; 2004. http://seclists.org/isn/2004/Feb/0011.html

[Senjyu] Senjyu, T.; Takara, H.; Uezato, K., Ito, To.; and Ito, Ta.; "Load-Following Control for Dispersed Generators Using Using a PI Controller"; *Electrical Engineering in Japan*, Vol. 145, No. 2, 2003. {Translated from Denki Gakkai Ronbunshi, Vol. 122-C, No. 8, August 2002, pp. 1333–1340}

[Thorp] Thorp, J.; Xiaoru Wang, X.; Hopkinson, K.; Denis V. Coury, D.; and Giovanini, R.; "Agent Technology Applied to the Protection of Power Systems"; *Proc. Conference on Securing Critical Infrastructures*; 2004

[Tolbert] Tolbert, L.; Qi, H.; and Peng, F.; "Scalable Multi-Agent System for Real-Time Electric Power Management"; *Proc. Power Engineering Society Summer Meeting*; 2001.

[Venkataramanan] Venkataramanan, G.; Illindala, M.; "Microgrids and sensitive loads"; *Proc. IEEE Power Engineering Society Winter Meeting, 2002*; Page 315; V 1, January 2002

[Wang] Wang, H.F.; "Multi-agent co-ordination for the secondary voltage control in power-system contingencies"; *IEEE Proc. on Generation, Transmission and Distribution;* 148(1): 61-66; 2001.

[Ygge95] Ygge, F. and Astor, E.; "Interacting Intelligent Software Agents in Demand Management"; *Proc. Distribution Automation and Demand Side Management '95*; 1995.

[Ygge96] Ygge, F.; Gustavsson, R.; and Akkermans, H.; HOMEBOTS: Intelligent Agents for Decentralized Load Management; *Proc. Distribution Automation and Demand Side Management '96*; 1996.

[Zambonelli] Zambonelli, F.; Jennings, N. R.; and Wooldridge, M.; "Developing Multiagent Systems: The Gaia Methodology"; in *ACM Transactions on Software Engineering Methodology*, 12(3):317-370, July 2003. http://polaris.ing.unimo.it/Zambonelli/PDF/ACMTOSEMaccepted.pdf

Appendix 1: Glossary

This section defines terms that appear in this document.

Admissible A plan is *admissible* if:

1) Its configurations are complete, consistent, and achievable in the real world.

2) All plan configurations and actions satisfy all constraints of the context in which the plan applies.

3) Each plan configuration (except the first) is a predictable consequence of the configuration that temporally precedes it as a result of processes believed to be active in that context and the actions, if any, that occur between the two configurations.

Agent An agent is a software program that presents the multi-agent system with a persona comprising one or more faces. Each face represents the agent's ability and authority to engage in a delimited set of activities in the process of achieving its organization's goals.

Cell A *cell* is a power system, comprising sources, loads, and a distribution network, that is simple enough to be managed by a single agent on a largely tactical basis.

Complete A configuration is *complete* if the power level and status of every device have values. A complete configuration could be the object of a command ("achieve this configuration"), a report ("at time t_0, the network was in the following configuration"), a forecast ("at time t_1, the network will be in the following configuration"), or an estimate ("network N will be in configuration C following event E")]

Configuration A *configuration* is the specification of power level, status, and operational settings for a given device or set of devices.

Consistent A configuration is *consistent* if the power levels, status, and operational settings of all devices in the configuration are equal, or nearly equal within specified limits, to the power levels and status of a real-world configuration with identical operational settings.

Distribution In the power domain, *distribution* lines are relatively low-voltage lines (12kv or less). A microgrid is primarily a distribution network. See also Transmission.

Entity An autonomous, situated locus of activity that makes decisions based on goals specific to itself. Agents (as discussed in this report), humans, corporations, and virtual loci made up of several interacting subordinate entities are all examples of entities.

Face A face is a set of roles grouped in support of a specific process (that require some set of functionality to exhibit).

Planner A piece of software that implements a planning algorithm.

Planning Domain The set of definitions of objects and actions that a planner uses to address planning problems.

Planning Problem A set of initial conditions and goal conditions, described in terms of a planning domain, between which a planner must find a path using the actions of the planning domain.

Role A *role* is a name given to a skill set that enables an organization to accomplish its goals. A role represents specific functionality that an agent can take on; an agent able to execute the functions can accept the role. Agents in designated roles interact to accomplish system tasks.

Transmission In the power domain, *transmission* lines are high-voltage lines (greater than 12kv) used to transmit power over long distances, in particular from a utility plant to high-voltage substations and distribution substations.

Virtual domain authority A group of entities enabled to act as if it were a single entity (hence the "authority figure" is virtual) for the purpose of authorizing activities within a given domain.

Appendix 2: Formal notation for liveness expressions

Operator	Interpretation
x,y	x followed by y
$x\|y$	x or y occurs
x^*	x occurs 0 or more times
$x+$	x occurs 1 or more times
x^ω	x occurs indefinitely often
$[x]$	x is optional
$x\|\|y$	x and y interleaved

Distribution

50	MS 0455	Laurence R. Phillips
2	MS 9018	Central Technical Files
2	MS 0899	Technical Library

— This page intentionally left blank —

Sandia National Laboratories