Multi-Agent Systems in a Distributed Smart Grid: Design and Implementation

M. Pipattanasomporn, *Member, IEEE*, H. Feroze, *Student, IEEE*, and S. Rahman, *Fellow, IEEE*

***Abstract*-- The objective of this paper is to discuss the design and implementation of a multi-agent system that provides intelligence to a distributed smart grid – a smart grid located at a distribution level. A multi-agent application development will be discussed that involves agent specification, application analysis, application design and application realization. The message exchange in the proposed multi-agent system is designed to be compatible with an IP-based network (IP = Internet Protocol) which is based on the IEEE standard on Foundation for Intelligent Physical Agent (FIPA). The paper demonstrates the use of multi-agent systems to control a distributed smart grid in a simulated environment. The simulation results indicate that the proposed multi-agent system can facilitate the seamless transition from grid connected to an island mode when upstream outages are detected. This denotes the capability of a multi-agent system as a technology for managing the microgrid operation.**

***Index Terms*--Distributed smart grid, multi-agent system and microgrid.**

I. INTRODUCTION

According to the U.S. Department of Energy (DOE)'s Modern Grid Initiative [1], a smart grid integrates advanced sensing technologies, control methods and integrated communications into current electricity grid – both at transmission and distribution levels. The smart grid is expected to exhibit the following key characteristics:

- self-healing,
- consumer friendly,
- attack resistant,
- provides power quality for 21st century needs,
- able to accommodate all generation and storage options,
- enables markets and
- optimizes assets and operates efficiently.

The focus of this paper is to discuss the design and implementation of a multi-agent system in the context of a distributed smart grid or a smart grid located at a distribution level. In particular, we will focus on implementing the concept of agents in an Intelligent Distributed Autonomous Power System (IDAPS) environment. IDAPS is a distributed smart grid concept proposed by Advanced Research Institute of Virginia Tech [2]. Having a built-in multi-agent functionality, IDAPS can be perceived as a microgrid that is intelligent. According to the U.S. DOE [3], a microgrid is defined as an integrated energy system consisting of interconnected loads and distributed energy resources which as integrated system can operate in parallel with the grid or in an intentional island mode.

Central to the operation of any power system is its control architecture consisting of hardware and software protocols for exchanging system status and control signals. In conventional electric power systems, this is accomplished by Supervisory Control and Data Acquisition (SCADA) systems [4,5]. Current trends to control and monitor the operation of electric power systems are however moving toward the use of an automated agent technology, which is generally known as a multi-agent system. A multi-agent system is a combination of several agents working in collaboration pursuing assigned tasks to achieve the overall goal of the system. The multi-agent system has become an increasingly powerful tool in developing complex systems that take advantages of agent properties: autonomy, sociality, reactivity and pro-activity [6]. The multi-agent system is autonomous in that they operate without human interventions. The multi-agent system is social-able in that they interact with other agents via some kind of agent communication language. The agents also perceive and react to their environment. Lastly, the multi-agent system is proactive in that they are able to exhibit goal-oriented behavior by taking initiatives.

In the context of power systems, multi-agent technologies can be applied in a variety of applications, such as to perform power system disturbance diagnosis [7,8], power system restoration [9], power system secondary voltage control [10] and power system visualization [11]. Some of the most recent work [12,13,14,15,16] has implemented a multi-agent system to control the operation of a microgrid. A report by Sandia [12] demonstrated that the agent-based technology can be used to control generators in a microgrid with a case study of a small military camp setting. The use of a multi-agent system to control a small microgrid that comprises PV generators, batteries and controllable loads is discussed in [13, 14, 15]. A multi-agent system approach for the restoration of a power system shipboard microgrid is discussed in [16], using case studies of a simulated shipboard power system.

This paper discusses the design and implementation of a multi-agent technology in the context of Intelligent Distributed

This work was supported in part by the U.S. National Science Foundation under Grant ECCS-0742832 and the U.S. Department of Defense under Grant W912HQ-08-C-0037.

M. Pipattanasomporn is with Virginia Tech – Advanced Research Institute, Arlington, VA 22203 USA (e-mail: mpipatta@vt.edu).

H. Feroze is with Virginia Tech – Advanced Research Institute, Arlington, VA 22203 USA (e-mail: hferoze@vt.edu).

S. Rahman is professor and director of Virginia Tech – Advanced Research Institute, Arlington, VA 22203 USA (e-mail: srahman@vt.edu).

Autonomous Power Systems (IDAPS). The agents in the IDAPS multi-agent system work in collaboration to detect upstream outages and react accordingly to allow the microgrid to operate autonomously in an island. This capability can be perceived as a software alternative to a traditional hardware-based zonal protection system for isolating a microgrid. The multi-agent system will provide a more flexible way that allows the redefinition of zonal boundary on the fly.

Section II briefly discusses the overview of an IDAPS microgrid. Section III presents the proposed multi-agent system architecture for use in the IDAPS microgrid. Section IV discusses how an IDAPS multi-agent system can be designed and implemented. This involves agent specification, application analysis, application design and application realization. Section V discusses the multi-agent implementation by demonstrating that the proposed multi-agent system can successfully control a simulated microgrid in real-time.

II. Overview of an IDAPS Microgrid

The relationship between generation, transmission and distribution sections of power systems and an IDAPS microgrid is illustrated in Fig. 1. Voltage levels as shown represent electric services that serve the campus of Virginia Tech in Blacksburg, VA. Virginia Tech Electric Service purchases electricity from American Electric Power at 69kV, which is then transformed to 12.47kV at campus substations. In this context, an IDAPS microgrid comprises all components of a distribution circuit at or below 12.47kV. The case study presented in section V is based on the system voltage and settings discussed in this section.

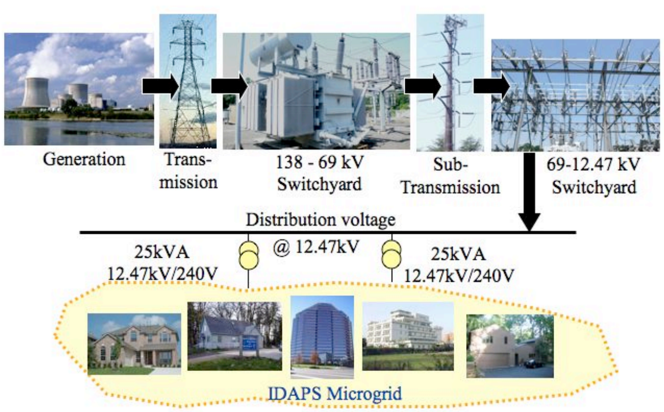

Fig. 1. The boundary of an IDAPS microgrid in a Virginia Tech setting

A typical operating strategy of an IDAPS includes both "normal operation" and "outage mode operation". During normal operating conditions, the IDAPS microgrid runs as a part of the local utility and coordinates its internal loads and distributed energy resources (DERs) for the most optimal operation. DERs of interest may include solar photovoltaics, wind turbines, microturbines, fuel cells and battery storage. In the event of an upstream outage, the IDAPS multi-agent control architecture is designed to isolate the IDAPS microgrid from the local utility. In such a situation, the IDAPS microgrid performs load controls based on a predefined prioritized list and activates its internal generators to secure critical loads. Key characteristics of an IDAPS microgrid are intelligent, distributed and autonomous, each of which is summarized below.

Intelligent: IDAPS performs demand-side management based on price signals; IDAPS secures critical loads, sheds low priority loads and allows for locally available power to be shared within a community.

Distributed: DER devices are dispersed in nature; various DER devices communicate within IDAPS through the IP-based local control model (IP = Internet Protocol).

Autonomous: IDAPS disconnects itself from the local distribution utility and operates autonomously to maintain the integrity of the system.

III. The Multi-Agent System Architecture in the IDAPS Microgrid

The idea behind any multi-agent system is to break down a complex problem handled by a single entity – a centralized system – into smaller simpler problems handled by several entities – a distributed system. The architecture of an IDAPS multi-agent system is presented in Fig. 2.

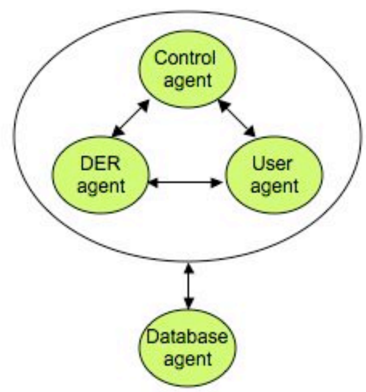

Fig. 2. The IDAPS multi-agent architecture, where arrows represent messaging exchange among agents via the Transmission Control Protocol/Internet Protocol (TCP/IP)

As shown, the IDAPS multi-agent system comprises four types of agent, namely a control agent, a DER agent, a user agent and a database agent. The proposed agent architecture is a variation of the agent architecture presented in [2], which comprises seven agents. The proposed agent architecture reduces the number of messages exchanged among agents and simplifies the overall complexity of a multi-agent implementation in an IDAPS microgrid.

In the IDAPS multi-agent system, each agent has unique objectives and responsibilities. When working in collaboration, four agents will work toward achieving the overall goal of an IDAPS microgrid, which is to secure critical loads within the microgrid during outages. Objectives and responsibilities of each agent will be discussed in the next section.

IV. The Multi-Agent System Design and Implementation

To implement a multi-agent system, there are a number of open-source agent platforms available in the literature that aid developers to build a complex agent system in a simplified fashion. These open-source agent platforms include: Aglets software development kit [17], Voyager [18], Zeus [19], JADE [20], Tracy [21, 22], SPRINGS [23] and SkeletonAgent [24]. Since performance of these agent development toolkits has been extensively studied in many previous publications [25, 26, 27, 28, 29, 30, 31], comparison of these agent platforms will not be discussed in this paper.

In the context of the IDAPS microgrid, it is very important to select an agent platform that is based on a well-known standard, that is the IEEE standard on Foundation for Intelligent Physical Agents (FIPA). This will help ensure interoperability among different systems and platforms so that the proposed multi-agent system can be universally accepted. As a result, compliance with FIPA should be the number one criterion for selecting an agent platform to be used in this task. Based on the agent toolkits listed above, agent platforms that are FIPA-compliance are Zeus and JADE. Comparison of these two agent platforms is presented in Table I below.

TABLE I
JADE and ZEUS Comparison

JADE	ZEUS
Free and open source	Free and open source
FIPA Compliant	FIPA Compliant
JAVA Based	JAVA Based
ACL Communication	ACL & KQML
Provides authentication	Some security capabilities
Decent GUI	User friendly GUI

Note: ACL = Agent Communication Language; KQML = Knowledge Query Manipulation Language (KQML); GUI = Graphic User Interface

In this study, Zeus is selected for the IDAPS implementation because Zeus has several user-friendly features that facilitate agent communications and allow negotiations for power exchange to be initiated easily and quickly. The philosophy behind Zeus development is that it hides intricacies of existing multi-agent systems and provides a relatively general purpose and customizable, collaborative agent building toolkit [19] that could be easily used by software engineers having only rudimentary knowledge in agent technology.

The IDAPS multi-agent application development in Zeus involves agent specification, application analysis, application design, application realization and application implementation. Each step is explained in more detail below. The application implementation is discussed in section V where a case study is presented.

A. Agent Specification

In this step, specifications of a control agent, a DER agent, a user agent and a database agent in the IDAPS multi-agent system are defined.

1. <u>Control agent</u> puts forth responsibilities that include monitoring system voltage and frequency to detect contingency situations or grid failures, and sending signals to the main circuit breaker to isolate the IDAPS microgrid from the utility when an upstream outage is detected. The responsibilities also include receiving electricity price ($/kWh) signal from the main grid, which may be obtained from advanced metering infrastructure, and publishing them to the rest of IDAPS entities.

2. <u>DER agent</u> is responsible for storing associated DER information, as well as monitoring and controlling DER power levels and its connect/disconnect status. DER information to be stored may include DER identification number, type (solar cells, microturbines, fuel cells, etc), power rating (kW), local fuel availability, cost function or price at which users agree to sell, as well as DER availability, i.e. planned maintenance schedule.

3. <u>User agent</u> acts as a customer gateway that makes features of an IDAPS microgrid accessible to users. It includes responsibility of providing users with real-time information of entities residing in the IDAPS system. A user agent also monitors electricity consumption by each critical and non-critical load. A user agent also allows users to control the status of loads based on priority predefined by a user.

4. <u>Database agent</u> is responsible for storing system information, as well as recording the messages and data shared among agents. Database agent also serves as a data access point for other agents, as well as users.

B. Application Analysis

After the agent specification step, the second step involves the formalization of agent roles and responsibilities that simplify understanding and modeling the problem at hand, using a role modeling technique.

To accomplish this task, a collaborative diagram – which defines the interaction among agents and their interaction with the environment – needs to be defined. The collaborative diagram of an IDAPS multi-agent system is shown in Fig. 3.

Fig. 3 illustrates three IDAPS agents – a control agent, a DER agent and a user agent – and their interactions with each other and the environment. All three IDAPS agents communicate with the IDAPS database agent, which comprises the name server and the facilitator. The visualizer receives copies of all messages exchanged within the multi-agent system and is responsible for displaying these messages to users. In Zeus, the name server, the facilitator and the visualizer are collectively termed utility agents. All messaging exchanges among agents are established via the Transmission Control Protocol/Internet Protocol or TCP/IP.

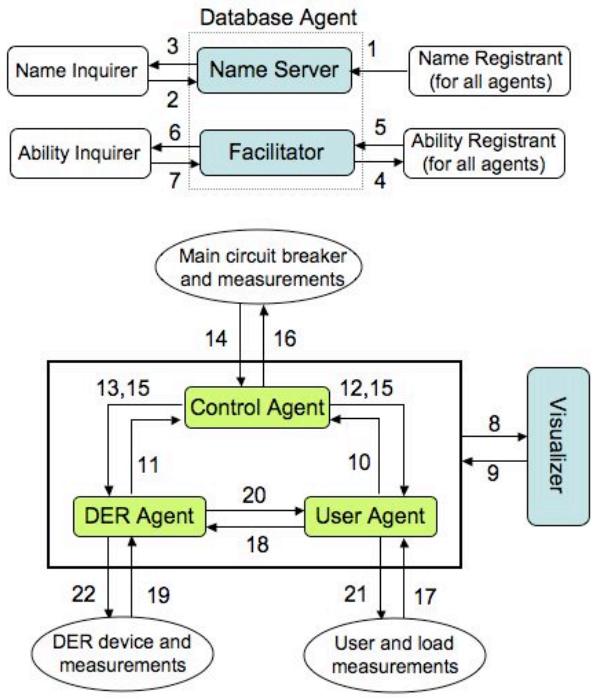

Fig. 3. IDAPS multi-agent system collaborative diagram

The initialization of the IDAPS multi-agent system is performed by the control agent, the DER agent and the user agent notifying the database agent of their presence. This includes (i) all agents notifying the name server of their names and IP addresses; the name server then updates its database and fulfills the required procedure for agent registration (1-3); (ii) all agents notifying the facilitator of their abilities; the facilitator then fills its database with information regarding abilities of agents (4-7). After the initialization process, the user agent and the DER agent register with its associated control agent (10-11). The control agent responds to the registration requests (12-13).

With respect to the interaction of agents with their environments, the control agent receives measurements from the main grid and publishes that information to its registered agents (14-15). If the upstream outage is detected, the control agent sends out a control signal to the main circuit breaker (16). The user agent receives power requirement information from the loads and sends command to the DER agent (17, 18). On the other hand, the DER agent receives power production information from the DER and sends out this information to the user agent (19, 20). The user agent and the DER agent react to their environments (21, 22) according to rules predetermined by a user. The visualizer collects any copy of messages being exchanged among all agents (8-9) so that they can be further displayed.

Note that the discussion above refers to interactions among agents in a single IDAPS MAS. More than one multi-agent systems can be created using the same collaborative diagram presented above.

C. Application Design

The application design step involves a process of modeling knowledge that will be used by each agent. In this step, *Facts* are defined for the application. *Facts* represent statements that an agent believes to be true, either about itself or its external environment [19]. Table II gives a general idea about how these *Facts* are defined.

TABLE II
FACTS IN THE IDAPS MULTI-AGENT SYSTEM

Facts	Value
Islandmode	True/False
Energy	kWh
Load circuit breakers	Close/Open
DG Operational cost	cents/kWh
Real-time electricity price	cents/kWh
Power requirement at loads	kW
Power produced by DGs	kW
Status of DGs	On/Off

In addition to modeling knowledge as *Facts*, application design also involves mapping agent responsibilities to the problems that each agent attempts to solve. This paper focuses on the social and domain responsibilities of the control agents, as well as those of the user and DER agents. The database agent is created automatically in Zeus, the social and domain responsibilities of the database agent will therefore not be discussed.

<u>Social & domain responsibilities of the control agent and the application design:</u>

- Accept incoming registration: add fact monitor, i.e. when new facts are received, appropriate action can be taken.
- Respond to registrations: create an initial agent resource and implement a message sending rule.
- Monitor system voltage: add fact monitor, i.e. when information is received, a fact is added.
- Update subscribers: implement message sending rules, i.e. when the rule is triggered, the control agent can send a message to the user and DER agents.
- Interpret main grid state: implement an adapter method which provides the application specific code that gives the rules for interpreting current state of the circuit.
- Store information: store information in the resource database, in an external database, or in Java code that is accessible to the agent's ZeusExternal interface.
- Provide a graphical depiction: implement GUI in the agent's ZeusExternal program.

<u>Social & domain responsibilities of the DER and user agents and their application design:</u>

- Register with the control agent: use the facilitator and implement message sending rules.
- Communicate with other user or DER agents: use the facilitator and implement message sending rules
- Sense and react to the external measurements: implement behavior rule base and store the expertise required to react to the change in circuits in the form of rules.
- Update the user on current progress: implement external

program, such as GUI.

- Receive user inputs: implement external program, such as GUI that takes user inputs.

D. Application Realization

The application realization stage consists of several steps required to create a generic agent capable of performing roles identified in previous sections. The application realization process consists of ontology creation, agent creation, utility agents configuration, agent task configuration and agent code generation.

Ontology is the vocabulary used by agents during communications. Ontology is created for the multi-agent system based on the application design process mentioned in the previous section. *Facts*, constraints, types, attributes and default values are put together as an environment model for the multi-agent system. In the ontology creation step, *Facts* are classified into two types, namely abstract and entity. In the IDAPS multi-agent system, the energy that is bought or sold is only one entity fact. The rest of the facts are abstract facts.

The agent creation process involves the creation of a generic agent of each type (control agent, user agent, DER agent). This is done by attributing certain tasks, rules, co-ordination protocols and status to each agent. In the IDAPS multi-agent development, a rules-based approach has been selected, that is, agents act in a reactive manner to their environment according to a set of predefined rules. FIPA compliant co-ordination protocols are attributed to all agents. All agents are considered to be peers i.e. at the same status in the application.

In the utility agent configuration step, the database agent – including the name server and the facilitator – as well as the visualizer, are configured. In the task agent configuration step, generic agents from the agent creation step are linked to external programs or java classes. This allows querying and modifying agent internal states, as well as interaction between users and the application. Finally agent codes are generated which complete the application realization process.

V. Implementation of the Proposed Multi-Agent System in a Simulated Microgrid

The objective of this simulation is to demonstrate that the proposed multi-agent system can facilitate the seamless transition of the IDAPS microgrid from the grid connected operation to the island mode when an upstream outage is detected. The simulation setup, the case study description and results are discussed below.

A. Components of the Simulated Circuit

In order to implement the proposed multi-agent system, a simulation test bed is developed in Matlab/Simulink as a simplified distribution circuit that comprises: a 50kW distributed generator, grid interface (inverter, pulse width modulation or PWM controller, low-pass filter and isolation transformer), loads (including 50kW critical loads and 12.5kW non-critical loads) and load circuit breakers which are IP-enabled, a distribution transformer (12.47kV/240V), the main circuit breaker and the utility grid at 12.47kV. See Fig. 4. A fault is applied to the upstream circuit at 0.1 second after the simulation starts.

Measurements are voltage and current waveforms at bus B2, which represent voltage and current outputs produced from the distributed generator after having been converted to 60Hz AC signals.

B. TCP/IP connections between the Simulated Circuit and the External Multi-Agent System

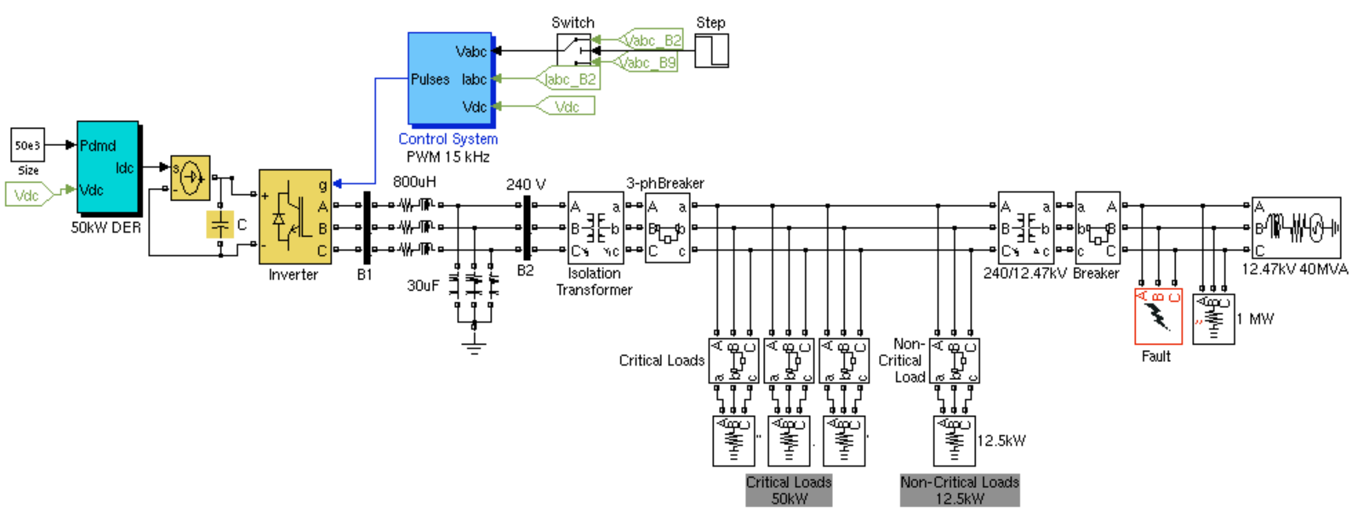

Fig. 4. Microgrid hardware simulation in Matlab, consisting of a 50kW distributed generator, grid interface, loads and load circuit breakers, a distribution transformer (12.47kV/240V), the main circuit breaker and the utility grid at 12.47kV

The simulation is set up in such a way that the simulated microgrid – as shown in Fig. 4 – resides in one computer (computer 1) and the multi-agent system resides in the other computer (computer 2). See Fig. 5.

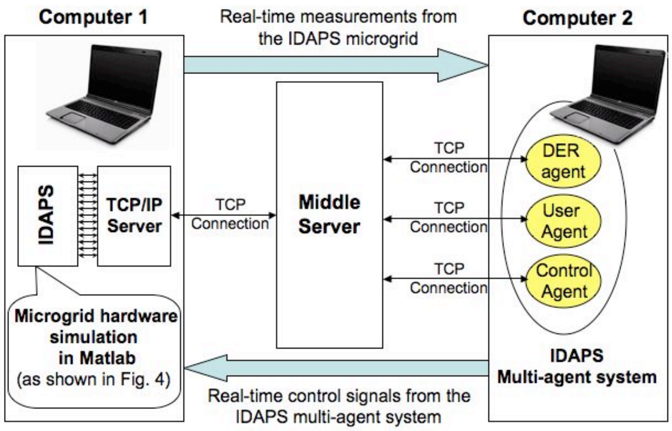

Fig. 5. Simulation setup

The IDAPS multi-agent system is connected to the simulated microgrid via TCP/IP connections. According to this setup, the microgrid hardware simulation in Matlab can be controlled by the multi-agent system from any remote location.

Real-time measurements and control signals are exchanged between these two computers as follows:

- Real-time measurements from computer 1 – including voltages, currents, status of circuit breakers, power produced by the distributed generator and power consumption by critical and non-critical loads – are sensed and monitored by the IDAPS multi-agent system in computer 2 via TCP/IP.
- The IDAPS multi-agent system in computer 2 issues real-time control signals to control the loads connected through the IP-enabled circuit breakers, as well as the power output from the distributed generator within the simulated microgrid in computer 1, also via TCP/IP.

A third party TCP/IP server [32] is implemented in Matlab/Simulink to establish TCP connectivity between the microgrid in the simulated environment and the multi-agent system residing in the other computer. The TCP/IP server is capable of conveying 13 real-time control signals and forwarding unlimited number of real-time circuit measurements from and to the external multi-agent system, respectively. However, it only allows a single TCP/IP connection to an external system at a time. As a result, a middle server is developed to allow multiple TCP connections to the external multi-agent system. In this case, the middle server allows multiple agents, i.e. control agent, user agent and DER agent, to connect to the simulated circuit simultaneously.

C. Results and Discussion

The circuit as shown in Fig. 4 is simulated for 0.2 second. The simulation result shown in Fig. 6 illustrates the variation of 60Hz voltage and current waveforms – measured at bus B2 – in per unit, before and after the upstream outage applied to the circuit at 0.1 second. Note that the per unit base power is 50kW and the per unit line-line base voltage is 240V.

As can be seen from Fig. 6, the circuit is in the grid-connected mode during the first 0.1 second and it is in the island mode thereafter.

During the grid-connected mode: the microgrid's voltage and frequency are controlled such that they follow the grid's voltage and frequency, which are roughly at 1 per unit and 60 Hz, respectively. The total microgrid demand is 62.5kW, including 50kW critical loads and 12.5kW non-critical loads. During the grid-connected mode, the IDAPS microgrid produces 25kW internally. This can be seen from the measurements in Fig. 6, i.e. the voltage output is 1 per unit and the current output is 0.5 per unit. This is equivalent to 0.5 per unit or 25kW of power. Thus, an additional 37.5kW from the grid is required to supply the overall microgrid demand.

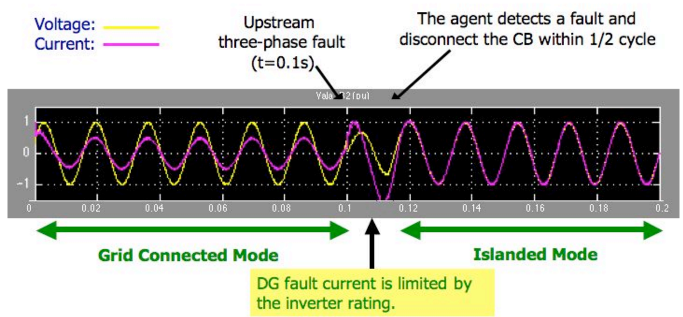

Fig. 6. Real-time simulation result (Y-axis represents voltage and current waveforms in per unit measured at bus B2. See Fig. 4; X-axis is the simulation time in second).

During the transition: Once the control agent detects the fault at t = 0.1, the control agent informs the user agent and the DER agent, both of which exchange information and determine the amount of loads to be shed and the amount of power to be produced internally in order to stabilize the microgrid. Depending on the pre-defined load priority set by a user and the available internal generation, the user agent can disconnect non-critical loads and certain critical loads. In this case, it is assumed that the generator can produce power at its full capacity, i.e. 50kW, without any fuel limitation during the outage. Thus, the user agent only disconnects the non-critical loads of 12.5kW and leaves the critical load intact. The control agent also sends a control signal to isolate the microgrid by disconnecting the main circuit breaker as soon as the upstream outage is detected. All agent actions – from detecting the fault, disconnecting the main circuit breaker, disconnecting the non-critical loads to stabilizing the grid – can be accomplished within half an electrical cycle, i.e. less than 0.008 second for a 60-Hz system.

The DER fault current as shown is limited by the inverter rating in the case of an inverter-based DER. The implication of the size of the fault current on the microgrid operation depends upon how the existing protection system is modified to accommodate the penetration of distributed energy resources in the distribution circuit. This can be studied as a future work and will not be discussed herein.

In the island mode: after the fault occurred at 0.1 second, the IDAPS microgrid is disconnected from the main grid. When the microgrid is operated in the island mode, the voltage is always controlled at 1 per unit and the frequency is always controlled at 60Hz. The user agent and the DER agent balance the demand and supply within the microgrid. The total system load is reduced to its critical loads of 50kW and the distributed generator produces 50kW internally to supply these critical loads. As seen in Fig. 6, while the voltage is always kept constant at 1 per unit, the current increases from 0.5 per unit to 1 per unit during the transition. This implies that the power output of the IDAPS internal generator increases from 25kW to 50kW after the upstream outage is detected.

VI. Conclusion

This paper discusses the design and implementation of the multi-agent system for use in an IDAPS microgrid. The proposed multi-agent system consists of a control agent, a DER agent, a user agent and a database agent. Agents exchange their messages via a TCP/IP protocol based on the IEEE FIPA standard to ensure the system interoperability. The application implementation process is illustrated through a real-time simulated case study, which indicates that the proposed multi-agent system can disconnect and stabilize the microgrid from the local utility when upstream outages are detected. This illustrates the capability of a multi-agent system as a technology for managing the microgrid operation. Multi-agent system's timely response facilitates the seamless transition from grid connected to island mode on detection of upstream outages in a microgrid. This demonstrates that the agent's capability can be considered as a software alternative to a traditional hardware-based zonal protection system for isolating a microgrid. Therefore, the multi-agent system provides a more flexible and updatable IDAPS layout, which will allow the redefinition of the microgrid zonal boundary on the fly. In addition to serving as a flexible protection alternative, the IDAPS multi-agent system also sheds non-critical loads according to a pre-defined prioritized list while stabilizing the microgrid after its isolation from the main grid. In sum, this work aims at demonstrating a practical implementation of multi-agent systems in a smart grid located at a distribution level.

VII. References

[1] Modern grid initiative: http://www.netl.doe.gov/moderngrid/.

[2] S. Rahman, M. Pipattanasomporn and Y. Teklu, "Intelligent Distributed Autonomous Power Systems (IDAPS)", *In Proc. 2007 the IEEE PES Annual General Meeting*, Tampa, Florida, 8pp.

[3] P. Agrawal, "Overview of DOE Microgrid Activities", Symposium on Microgrid, Montreal, June 23, 2006 [Online]. Available: http://der.lbl.gov/2006microgrids_files/USA/Presentation_7_Part1_Poonum-agrawal.pdf.

[4] R. Krutz, *Securing SCADA systems*. Wiley Publishing, Inc. 2006.

[5] D. Bailey and E. Wright, *Practical SCADA for industry*. Newnes, Oxford, UK, 2003.

[6] S. Franklin and A. Graesser, "Is it an agent, or just a program?: a taxonomy for autonomous agents", In Proc. 1996 the 3rd International Workshop on Agent Theories, Architectures and Languages, Springer-Verlag, pp.21-35.

[7] J. Hossack, S.D.J. Mcanhur, J.R. Mcdonald, J. Stokoe and T. Cumming, "A multi-agent approach to power system disturbance diagnosis", In Proc. International conference on power system management and control, April 2002, Vol. 488, pp. 317-322.

[8] S.D.J. McArthur, E.M. Davidson, J.A. Hossack, and R. McDonald, "Automating power system fault diagnosis through multi-agent system technology", *In Proc. 2004 the 37th Hawaii International Conference on System Sciences*, 8pp.

[9] T. Nagata and H. Sasaki, "A multi-agent approach to power system restoration", IEEE Transactions on Power Systems, May 2002, Vol. 17, pp. 457-462.

[10] H.F. Wang, "Multi-agent co-ordination for the secondary voltage control in power system contingencies", *In Proc. IEE Generation, Transmission and Distribution*, Jan 2001, Vol. 148, pp. 61-66.

[11] L. Cristaldi, A. Monti, R. Ottoboni and F. Ponci, "Multi-agent based power systems monitoring platform: a prototype", *In Proc. IEEE Power Tech Conference*, June 2003, Vol. 2, 5pp.

[12] L. Phillips, M. Link, R. Smith and L. Weiland, "Agent-based control of distributed infrastructure resources", Sandia National Laboratories, SAND2005-7937, Jan 2006 [Online]. Available: www.sandia.gov/scada/documents/sand_2005_7937.pdf.

[13] A. Dimeas and N.D Hatziargyriou, "A MAS architecture for microgrid control", *In Proc. 2005 the 13th International Conference on Intelligent Systems Application to Power Systems*, November, 5pp.

[14] A. Dimeas and N.D Hatziargyriou, "Operation of a multiagent system for microgrid control", IEEE transaction on power systems, Vol. 20, No. 3, August 2005. pp. 1447-1455.

[15] A. Dimeas and N.D Hatziargyriou, "A multi-agent system for microgrids", *In Proc. 2004 IEEE Power Engineering Society General Meeting*, Vol. 1, pp. 55-58.

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

[17] Aglets agent development toolkit [Online]. Available: http://aglets.sourceforge.net/.

[18] Voyager agent development toolkit [Online]. Available: http://www.recursionsw.com/.

[19] Zeus agent development toolkit [Online]. Available: http://labs.bt.com/projects/agents/zeus/.

[20] JADE agent development toolkit [Online]. Available: http://jade.tilab.com.

[21] Tracy agent development toolkit [Online]. Available: www.mobile-agents.org.

[22] P. Braun and W. Rossak, *Mobile Agents: Basic Concepts, Mobility Models and the Tracy Toolkit*, Morgan Kaufmann, 2004.

[23] Springs agent development toolkit [Online]. Available: http://sid.cps.unizar.es/SPRINGS.

[24] D. Camacho, R. Aler, D. Borrajo and J. Molina, "Agent architecture for intelligent gathering system", AI Communications, Vol. 18, Iss. 1, pp. 15-32, January 2005.

[25] E. Shakshuki, and Y. Jun, *Multi-agent Development Toolkits: An Evaluation*, Springer Berlin/Heidelberg, 2004.

[26] R. Trillo, S. Ilarri and E. Mena, "Comparison and Performance Evaluation of Mobile Agent Platforms", *In Proc. 2007 the Third International Conference on Autonomic and Autonomous Systems (ICAS'07)*, Athens, Greece.

[27] D. Camacho, R. Aler, C. Castro and J. Molina, "Performance evaluation of Zeus, JADE and SkeletonAgent Framework", *In Proc. 2002 IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, 6 pp.

[28] M.K. Perdikeas, F.G. Chatzipapadopoulos, I.S. Venieris and G. Marino, "Mobile agent standards and available platforms", Computer Networks, vol. 31, no. 19, pp. 1999–2016, August 1999.

[29] M.D. Dikaiakos and G. Samaras, "Performance evaluation of mobile agents: Issues and approaches", Lecture notes in computer science Vol. 2047. Springer-Verlage, May 2001, pp. 148– 166.

[30] A.R. Silva, A. Romao, D. Deugo and M. Silva, "Towards a reference model for surveying mobile agent systems", Autonomous Agents and Multi-Agent Systems, vol. 4, no. 3, pp. 187–231, September 2001.

[31] E. Shakshuki, "A methodology for evaluating agent toolkits", *In Proc. 2005 IEEE International Conference on Information Technology: Coding and Computing (ITCC'05)*, IEEE Computer Society, pp. 391-396.

[32] W. Zimmermann, "IOlib – Hardware Input/Output Library for Matlab and Simulink" [Online]. Available: http://www.it.hs-esslingen.de/~zimmerma/software/index_uk.html.

VIII. BIOGRAPHIES

Manisa Pipattanasomporn (S'01, M'06 – IEEE) joined Virginia Tech's Department of Electrical and Computer Engineering as an assistant professor in 2006. She received her Ph.D. in electrical engineering from Virginia Tech in 2004. She received the M.S. degree in Energy Economics and Planning from Asian Institute of Technology (AIT), Thailand in 2001 and a B.S. degree from the Electrical Engineering Department, Faculty of Engineering, Chulalongkorn University, Thailand in 1999. She is currently researching the application of a specialized microgrid called the Intelligent Distributed Autonomous Power Systems (IDAPS) to improve the resiliency of electrical energy infrastructures. Her fields of interest are renewable energy systems, distributed energy resources and critical infrastructures.

Hassan Feroze (S'07 – IEEE) is pursuing his M.S. degree in the Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University, VA, USA. He received his B.S. degree in Electrical Engineering from University of Engineering and Technology (UET), Lahore, Pakistan in 2005. His employment experiences include Qualcomm Inc., Virginia Tech's Advanced Research Institute and UET's Al-Khawarizmi Institute of Computer Science. He is currently member of the research group working on application of the Intelligent Distributed Autonomous Power Systems (IDAPS). His fields of interests include multi-agent systems implementation in power systems, and wireless and computer networks.

Saifur Rahman (S'75, M'78, SM'83, F'98 – IEEE) is the director of the Advanced Research Institute at Virginia Tech where he is the Joseph Loring Professor of electrical and computer engineering. He also directs the Center for Energy and the Global Environment at the university. Professor Rahman has served as a program director in engineering at the US National Science Foundation between 1996 and 1999. He has served on the IEEE PES Governing Board as VP of industry relations, and VP of publications between 1999 and 2003. In 2006 he served as the vice president of the IEEE Publications Board, and a member of the IEEE Board of Governors. In 2008 he is serving as the vice president for New Intiatives and Outreach for the IEEE Power & Energy Society and a member of its Board. He is a member-at-large of the IEEE-USA Energy Policy Committee. He is a distinguished lecturer of IEEE PES, and has published over 300 papers on conventional and renewable energy systems, load forecasting, uncertainty evaluation and infrastructure planning.