EPOCHS: A Platform for Agent-Based Electric Power and Communication Simulation Built from Commercial Off-The-Shelf Components

Kenneth Hopkinson, *Member*, IEEE, Xiaoru Wang, *Member*, IEEE, Renan Giovanini, *Student Member*, IEEE, James Thorp, *Fellow*, IEEE, Kenneth Birman, Denis Coury, *Member*, IEEE

***Abstract*—This paper reports on the development and subsequent use of the *E*lectric *P*ower and *C*ommunication Sync*h*ronizing *S*imulator (EPOCHS), a distributed simulation environment. Existing electric power simulation tools accurately model power systems of the past, which were controlled as large regional power pools without significant communication elements. However, as power systems increasingly turn to protection and control systems that make use of computer networks, these simulators are less and less capable of predicting the likely behavior of the resulting power grids. Similarly, the tools used to evaluate new communication protocols and systems have been developed without attention to the roles they might play in power scenarios. EPOCHS integrates multiple research and commercial off-the-shelf (COTS) systems to bridge the gap.**

***Index Terms*—Agents, COTS Integration, Power Systems Evaluation, Networking, Simulation**

I. Introduction

The restructuring of the electric power system, the creation of competitive markets, and the introduction of new regulatory mechanisms are now well-established trends. Understanding how the electric power grid operates, and in particular how to monitor and control it, are necessary preconditions to achieving reliability and fairness in the presence of these changes. Yet, electric power simulators today do not model the network communication patterns seen in modern protection and control systems.

Traditional protection systems base decisions on local measurements, and employ control systems, which operate over rather slow, predictable communication systems. It has not been necessary to simulate communication in order to accurately model these electric power systems. However, over the last decade, power systems have begun to operate close to their transmission, generation, and stability limits. Protection and control systems are being placed under a correspondingly greater strain. Power engineers have begun to conclude that the use of network communication based on Internet standards is a natural choice to improve on the protection and control systems that are currently deployed by allowing them to gather data more quickly and over larger geographic areas.

It is natural to assume that electric power protection and control systems with more data, and operating over faster communications networks, will be more effective than their predecessors. However, the Internet was not designed for safety- and time-critical applications, and layering the needed mechanisms over Internet protocols such as TCP/IP is a potentially impossible undertaking. New evaluation tools are needed that allow electric power engineers to accurately model the electric power and communication conditions that will be encountered. A crucial evaluation method, which has not existed until now, is a tool to provide high-quality simulations of electric power scenarios while simultaneously modeling the behavior of computer communications protocols in realistic networks confronted with realistic scenarios, including load surges, outages, and other forms of dynamic stress. Realistic simulators allow electric power engineers to resolve problems in the laboratory, avoiding potentially costly or damaging mishaps in the field.

This article presents the **E**lectric **Po**wer and **C**ommunication Sync**h**ronizing **S**imulator (EPOCHS), a combined simulation system, or federation, which links the PSCAD/EMTDC electromagnetic transient simulator, the PSLF electromechanical transient simulator, and the Network

Manuscript received April 30, 2005. The authors were supported, in part, by DARPA under AFRL grant RADC F30602-99-1-0532 and by AFOSR under MURI grant F49620-02-1-0233. They were also supported by FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo, Brazil) and CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brazil). The views expressed in this document are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government. An early version of this paper appeared as, K. Hopkinson, X. Wang, K. Birman, J. Thorp, "EPOCHS: Integrated Commercial Off-The-Shelf Software for Agent-based Electric Power and Communication Simulation," Proceedings of the 2003 Winter Simulation Conference, Piscataway, NJ, 1158-1166. This paper extends that work by adding additional details about the EPOCHS platform and the special protection system use case.

K. Hopkinson is with the Department of Electrical and Computer Engineering, Air Force Institute of Technology, 2950 Hobson Way, Wright-Patterson AFB, OH USA (e-mail: kenneth.hopkinson@afit.edu).

X. Wang is with the School of Electrical Engineering, Southwest Jiaotong University, Chengdu, China (e-mail: xrwang@home.swjtu.edu.cn).

R. Giovanini and D. Coury are with the Department of Electrical Engineering, University of São Paulo at São Carlos, São Carlos, SP, Brazil. (e-mail: renan@sc.usp.br; coury@sc.usp.br).

J. Thorp is with the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute, Blacksburg, VA USA (e-mail: jsthorp@vt.edu).

K. Birman is with the Department of Computer Science, Cornell University, Ithaca, NY USA (e-mail: ken@cs.cornell.edu).

Simulator 2 (NS2) communication simulator. EPOCHS allows users to investigate electromagnetic scenarios involving communication, using PSCAD/EMTDC and NS2, or electromechanical scenarios involving communication, using PSLF and NS2. The technology underlying EPOCHS illustrates how non-intrusive techniques can be used to federate simulation engines using only their built-in application programming interfaces (APIs). EPOCHS' agent-based framework also hides the complexity involved in the combined simulation system, making it easy for users to design new power scenarios involving communication.

This paper is divided into eight sections. Section 2 reviews background material. Section 3 presents the system's architecture. The methods used to federate the commercial and research simulation components are discussed in section 4. Section 5 outlines EPOCHS' agent framework. Section 6 presents an example of EPOCHS' use. Section 7 outlines areas for future work. The article concludes in section 8.

II. Background

Constructing a new combined simulation engine is potentially time-consuming and expensive. This is particularly true when simulations have both continuous and discrete-event components. An alternative is to link multiple simulations into a distributed environment (federation). Using this approach to combine multiple simulators for use in inter-domain situations is becoming more common, and there are even proposals to standardize such architectures, notably the High Level Architecture (HLA) [1]. Commercial off-the-shelf simulation (COTS) systems are popular in many fields due to their rich feature sets, ease of use, and cost effectiveness. However, source code is only rarely available, and this stands as an obstacle to federation. In EPOCHS, source code was available for NS2 but not for PSCAD or PSLF.

Prior research on federating COTS simulators includes work on supply chain manufacturing simulation in the GRIDS project [2], Tucci and Revetria's HLA compliance project [3], and Strassburger's SLX federation [4]. In addition, NIST has developed the Distributed Manufacturing System (DMS) Adapter, a mechanism for distributed simulation similar to the HLA, but with a more manageable level of complexity, which is targeted towards the manufacturing community [5]. Despite these successes, the documented use of commercial simulation systems in federations is still relatively rare. A common concern is that "casual" modelers may find the added complexity difficult to manage in a federated simulation.

It should be stressed that many quality power and networking simulators exist. and that he ones chosen were not the only possible selections. Three alternative packages that have the capability to simulate electromechanical and electromagnetic transients include EMTP-RV [6], ATP [7], and NETOMAC [8]. OPNET [9] and QualNET [10] are commercial packages that are widely used in network simulations. It is likely that any of these packages could be integrated into the EPOCHS environment.

Many real-time power simulators exist. Real-time power simulators include RTDS [11], HYPERSIM [12], which integrates with Matlab and Simulink [13], and ARENE [14]. These simulators open the interesting possibility of testing real-time power network simulations either with real networks or with network simulations that can operate in real-time such as QualNet [10]. Real-time simulation is not a goal of the EPOCHS project, however. Instead, the focus of EPOCHS is to integrate power and network communication simulators so that their internal simulation timeclocks advance seamlessly.

The power of agents is broadly recognized and, as a result, a wide range of simulations make use of them. There are two main classes of agent-based simulators. The first uses agents as a mechanism for combining simulators. The flexibility agents provide can be used to more efficiently link simulation components, as illustrated in a paper by Wilson [15]. The second class of simulators use agents to model entities within a simulated world. Lee employed a mix of continuous and discrete-time simulators in an air-traffic control simulation using agents to represent key entities such as aircraft [16]. EPOCHS uses a similar approach to the one used by Lee.

The combined simulation of electric power and communication systems is rare in the literature. Papers that discuss the subject tend to either focus on system models without including simulated cases [17], include simulations that center on communication performance while assuming consistent power system behavior once communication arrives [18], or include power system examples that either assume constant network delays or provide real data transmitted over predictable dedicated lines [19]. One exception is work performed by a group at North Carolina State University [20]. The group created an interface between PSCAD/EMTDC and a module written in Java to allow agents to model behavior in a power system and to communicate over a network. Communication was modeled with a custom-developed Java module that was limited to a basic emulation of a token-ring network. The technique used to bridge their module with the EMTDC simulator is similar to the one used in EPOCHS, but N.C. State's basic network simulation limits its utility.

III. EPOCHS

A. Architecture

The HLA is an IEEE standard method [21] to combine individual simulations, known as federates, together into combined simulators known as federations. Recent work centering on federating simulation systems has focused on the use of the HLA. Examples of this can be found in domains such as artillery simulation [22], in robot simulation frameworks for military and manufacturing applications [23], and in manufacturing simulation systems [24]. Each of the simulation federates conforms to the HLA's rules, interface specifications, and documentation standards. The "glue" holding HLA combinations together is a central component known as a Runtime Infrastructure (RTI). The RTI routes all messages between simulation components and ensures that the

simulation time is synchronized across all components.

One drawback of the HLA is that it can be difficult to modify existing simulations to conform to its specification. Many fields make heavy use of off-the-shelf commercial software and do not have source code access. It is often possible to add a wrapper to these simulators to support their use in an HLA federation, but this is oftentimes problematic due to the number of additions that must be made to conform to the HLA, the limited input and output options that are present in many legacy simulation engines, and difficulties in accessing internal simulation state. Additionally, HLA can be an inefficient means of combining federates. The system is based on a publish-subscribe mechanism where any federate subscribing to another will receive all of its updated information, whether it is needed or not.

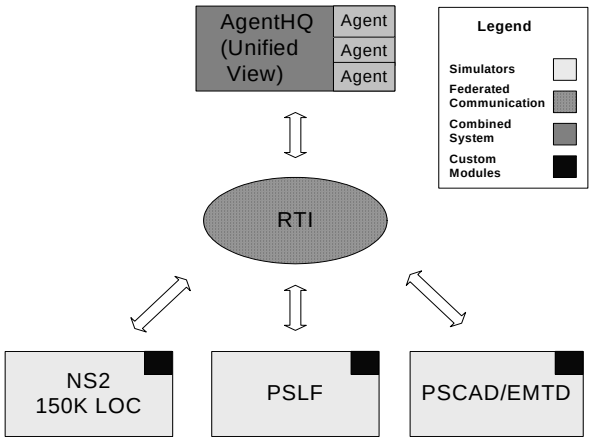


Fig. 1. The relationship between EPOCHS' Components. The RTI serves as an interface between all component simulators with EPOCHS. It ensures that the simulation clocks remain synchronized between all components and it also routes messages between them.

A federated simulation system was created that works in the spirit of the HLA, but which uses a custom interface for easier implementation. EPOCHS' architecture is shown in fig. 1.

It is critically important to choose the right simulation tools when creating a federated simulation system like EPOCHS. Each simulator used as a component in EPOCHS is viewed as one of the best within its class for some category of uses. Despite their advantages, none of the three simulation components (PSCAD, PSLF, and NS2) were designed for interoperability, posing a challenge that our work addresses.

Descriptions of the EPOCHS components follow.

PSCAD/EMTDC: PSCAD/EMTDC is used for electromagnetic transient simulation. EMTDC is a well-known electric power simulator. EMTDC models short-duration time-domain electric power responses. PSCAD is a graphical interface, which is used to simplify the development of EMTDC scenarios. PSCAD is produced by the Manitoba HVDC Research Centre [25].

EMTDC simulates power scenarios in continuous time by solving a series of differential equations in a time-stepped manner. It has very detailed electrical models, which makes it well-suited to electromagnetic transient investigations where the time-step size is often expressed in microseconds and representative situations last seconds or less.

PSLF: PSLF is used for electromechanical transient simulation. PSLF can simulate power systems with tens of thousands of nodes and is widely used by electric utilities to model electromechanical stability scenarios [26]. It models large systems in less detail than that available in PSCAD making it better-suited for long-running scenarios. PSLF simulates power systems in continuous time by solving differential equations in a time-stepped manner, which is similar to the method employed by PSCAD. PSLF's simulation engine is well-suited to investigations where the time-step is expressed in milliseconds and the total simulation time is expressed in minutes or less.

NS2: Network Simulator 2 (NS2) is an event-driven communication network simulator created through a joint effort between the University of California at Berkeley, Lawrence Berkeley Labs, the University of Southern California, and Xerox PARC. NS2 is a high-quality simulator, which allows the creation of a wide variety of communication scenarios. Although there are many network simulators, NS2 is the most widely used when evaluating the behavior of TCP, the TCP variants that have been proposed by researchers, and the behavior of routed UDP in large networks [27]. NS2 is able to simulate the behavior of these protocols under various forms of stress, such as might be caused by competition for network resources when multiple applications share a network and communicate over the same routers and communication links, the impact of failures including router failures, link failures, or denial of service attacks. It can also capture the normal dynamics resulting from relaying messages with real-time data rates through many layers of routers. NS2's detailed models of TCP make it particularly relevant to electric power situations involving communication since industry standards for the next generation of power equipment, such as the Utility Communications Architecture (UCA) 2.0 [28], make use of TCP running over Ethernet networks.

It must be emphasized that, while network simulators are generally able to replay traces of past network behavior, it is most common to use statistical communication models when running network simulations. Because network models are statistical, rather than fixed, it is vitally important to run a wide range of experiments under varying conditions before drawing conclusions from their results. EPOCHS depends on NS2 for the network aspects of all of its simulations so these considerations apply to results obtained from EPOCHS.

AgentHQ: AgentHQ presents a unified environment to agents and acts as a proxy when agents interact with other EPOCHS components. Through it, agents can get and set power system values, and can send and receive messages to each other. AgentHQ is a discrete-event system. Events are processed as they occur and are routed to the affected agents.

RTI: The Runtime Infrastructure (RTI) acts as the "glue" between all other components. It is responsible for simulation synchronization and for routing communication between EPOCHS' components. A firm requirement placed on any simulation system is that no event can be processed with a

time stamp earlier than one that has already been completed. This is easy to enforce in sequential simulators, but issues arise in distributed simulation systems. Many methods exist for dealing with this matter in the parallel and distributed simulation research community [29]. A time-stepped model, one of the simplest techniques for component synchronization, is used in EPOCHS. In the time-stepped model, each of the component federates executes until a preset simulation time is reached. When all of the federates has reached a target synchronization point, the federates are able to interact with one another by sending messages through the RTI. Once the federate interactions have ended, the RTI chooses a time for the next synchronization point, and all of the simulators continue execution. In our model, the amount of time between synchronization points has a fixed length. Time steps are user-selectable and can be chosen depending on the granularity of a given case. Simulations can use a short time step length to compensate for the errors introduced by the decoupled simulation approach or can use larger time steps for faster execution. It is important to recognize that the RTI synchronizes simulation time clocks rather than real time clocks. If simulator A takes 10 seconds for every 1 millisecond of simulation time and simulator B takes a half second per millisecond then these differences do not matter to the RTI. The important thing is that if an event occurs after five milliseconds of simulation time then it occurs at the same simulation time across all of EPOCHS' components.

B. Component Interaction

Synchronization between the simulators follows a simple algorithm. The network simulator NS2 and one of either the PSLF or PSCAD/EMTDC simulators will begin execution. The simulation systems are halted at time 0. At the beginning of any time step, the RTI waits for synchronization messages from both the power simulator and NS2. Then, the RTI yields control to the AgentHQ. The AgentHQ passes control on to the agents one by one until all have executed. During this cycle, the agents are capable of sending communication messages and getting/setting power system variables. Once all agents have executed, the AgentHQ returns control to the RTI. Finally, the RTI notifies both NS2 and the power system simulator that the current time step is done. Finally, the two simulation engines run for an additional time step.

Special attention must be paid to NS2. As a discrete-event simulator, messages may be received in between two synchronization points. This is not possible in the continuous-time PSLF and PSCAD/EMTDC components, which solve differential equations at fixed time steps. If a message arrives for an agent between synchronization points and the agent does not require electric power interaction then the agent proceeds normally. If the agent receives a message that requires that power system state be read or changed then the agent keeps the message in a queue until the next synchronization point. In this way, the interaction between the discrete-event NS2 simulator and the continuous-time PSLF or PSCAD/EMTDC simulators can lead to errors over time. Up to one time interval passes in error every time that a message requires an agent to act upon power system state.

The user has a dilemma when setting the interval between synchronization points in EPOCHS. Longer synchronization intervals will lead to faster simulations, but smaller intervals will decrease the cumulative system error. In the experiments run to date, care has been taken to select small time steps in order to minimize the error introduced by message events between simulation synchronization points.

IV. Federating COTS Components

Strassburger [4] lists four methods to make a simulator compliant with an RTI approach to simulation federation.

They are, from most to least desirable:

- Reimplement the tool with the proper extensions
- Extend the simulation with intermediate code
- Use an external programming interface
- Couple via a gateway program

In the first approach, developers modify a simulator's source code so that it can interface with the RTI. The second method is to include custom source modules to add RTI support if the simulator generates intermediate source code in a higher level language. Some tools allow users to call functions defined in custom source code or dynamic link libraries. The third option takes advantage of this facility to add RTI support. Finally, if none of these options exist, but a simulator includes facilities for external communication such as files, pipes, or sockets then the developer can communicate with an external gateway, which will serve as a proxy with the RTI. The federation of the three commercial and research systems serves as an interesting case study because each of them used a different technique from Strassbugers' list.

PSLF and PSCAD/EMTDC are commercial products and their source code is not available. NS2 is a research system with open source code, but it would take great effort to modify NS2's 150,000 lines of code to interface with an RTI. Instead, internal API's were used to federate each of the simulators.

EPOCHS' synchronization method is a convenient alternative to modifying the core source code in its component simulators. PSCAD/EMTDC, PSLF, and NS2 each allow user-defined extensions. A PSCAD/EMTDC scenario can include user-defined libraries that add equipment definitions, which were not present in the original software, using the C programming language. PSLF allows user-defined equipment models using its proprietary EPCL language. NS2 has a procedure for adding new communication protocols in C++. New equipment stubs have been created whose purpose is to interact with EPOCHS's RTI at each synchronization point. Both PSCAD/EMTDC and PSLF use a user-modifiable time step length. Both systems allow users to modify the time step length at will, but time steps were kept constant in the first EPOCHS release. Events are added to NS2 in order to ensure that it will also use a fixed length between each time step.

This process is simplified by the fact that PSCAD/EMTDC, PSLF, and NS2 are all single-threaded systems, so each

system is halted whenever a synchronization event occurs. Additional effort would be required if this were not the case.

The federation techniques employed in EPOCHS follow.

NS2: The network component uses Strassburger's second approach. A new transport protocol was added to NS2 to serve as its link to the RTI. A periodic call was added to the simulation script invoking the new protocol to halt execution and interact with the RTI once per time step. The length of the step is user-defined, but should be kept the same as that used in the power simulator. EPOCHS makes use of NS2's TCPApp application to track TCP/IP messages. NS2's UDP implementation has been extended to allow message tracking through a similar addition called UDPApp. These choices give users the flexibility to select any communication protocol at any time by sending data through NS2 function calls.

PSCAD/EMTDC: PSCAD/EMTDC uses the third federation method. PSCAD/EMTDC generates FORTRAN source code based on scenarios created in its graphical environment. Users can extend its functionality by making calls to source code in C or FORTRAN, which is compiled with the generated code. PSCAD/EMTDC is a continuous-time system. A library was created that adds a call to a custom component once per time step. The PSCAD/EMTDC component begins by reading in all equipment values that might be requested. Next, the component contacts the RTI and notifies it that the beginning of the time step has been reached. Agents can get or set equipment values when they execute. At the end of an execution cycle, a finish message is sent from the RTI to the electrical component and any values that changed are set in the simulator. Finally, control is passed back to the simulator and execution continues.

PSLF: In PSLF, the fourth federation method is used. PSLF includes its own language, EPCL, which is similar to C. Using EPCL, a stub was created that enabled interaction with the RTI through the use of files. All simulations are run on the same machine. If that were not the case then a gateway would be needed to act as a proxy when interacting with the RTI.

PSLF halts execution periodically and waits for requests from the RTI. Incoming requests to get or set electrical simulation values are processed and the results are returned to the RTI. When all requests have been fulfilled, a final message is sent to PSLF allowing it to continue execution

V. Agent Framework

Previous sections described the methods employed in federating EPOCHS' component simulators. A crucial piece missing from these discussions has been a description of the facilities provided to the users of the system to interact with it. In EPOCHS, users interact with the system through the use of agents. This section describes EPOCHS' agents, their capabilities, and the API provided for the system's users.

A. Agent Definition

The concept of "agents" is used extensively, but there is no universally accepted definition for the term. Agents nearly always have the properties of autonomy (the ability to take independent action) and interaction (the capacity to sense the surrounding environment and make changes to it). In addition, an agent may exhibit the properties of mobility, intelligence, adaptivity, and communication. In this paper, the term agent will be used to refer to computer programs that are autonomous, interactive, and have the ability to communicate over a network. Agents may optionally also have any of the other attributes defined above.

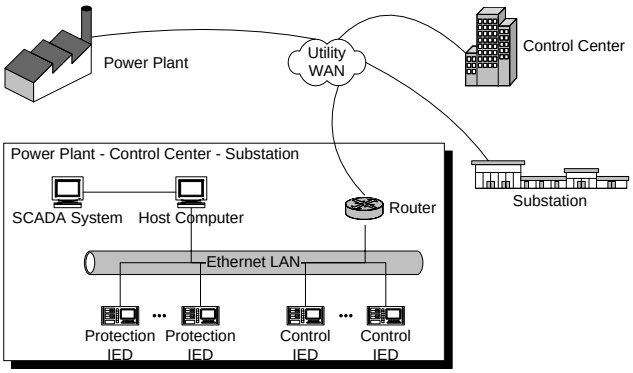


Fig. 2. Placement of the Agent-based IEDs within the Utility Intranet

B. Agents in Electric Power Protection and Control

The electric power grid has traditionally consisted of a large number of protection and control devices that act on local information to respond to problems. This method works well in some cases, but there are many situations in which information not readily available from local sensors or local databases would be needed to plan, to protect the grid, or to control it efficiently. Lacking such data, the grid must be operated in a very conservative and potentially inefficient manner, and might not be able to support desired behaviors. Agents have begun to be recognized as a natural way to introduce extensibility into the grid without drastically changing the usual power system architecture, and are therefore gaining acceptance in the electric power research community. Their autonomous nature, ability to share information and coordinate actions, and the potential to be easily upgraded or controlled from a remote location are appealing to grid operators and protocol designers.

The protection and control scenarios that interest us use geographically distributed agents located in a number of Intelligent Electronic Devices (IEDs), as shown in fig. 2. An IED is a hardware environment that has the necessary computational, communication, and I/O capabilities needed to support a software agent. An IED can be loaded with agents that can perform control and/or protection functions. These agent-based IEDs work in an autonomous manner, interacting both with their environment and with each other. For example, a digital relay could be implemented as an agent with its own thread of local control, but with the added ability to monitor conditions elsewhere in the network so as to act in response to non-local events. This agent would communicate with other agents either via Local Area Networks (LANs) or via Wide Area Networks (WANs). These IEDs are relatively rare at present, but many are already available and it is expected that their use will increase over time with the passage of the IEC

61850 standard for electric power equipment interoperability.

The IEC 61850 standard [30] builds upon the Utility Communications Architecture (UCA) [28] to create a method for electric power substations, IEDs, and other apparatus to communicate with one another. IEC 61850 specifies the types and names of data that will be available, the methods to access and exchange data, and the interfaces between power equipment and communication networks. IEC 61850 is divided into 14 parts, 10 of which have been published as of the time of this writing. Devices compliant with UCA 2.0 have been available since the year 2000. It is anticipated that equipment compliant with IEC 61850 will be available when the entire standard has been ratified. The important thing to note about the standard is that the power equipment specified, particularly the IEDs, incorporate microprocessors, memory, and communication systems. These IEDs allow agents and other pieces of software to be loaded into the equipment as envisioned in the definition of agents previously given.

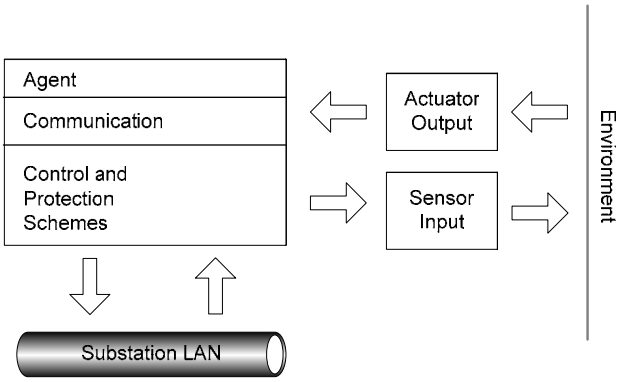


Fig. 3. The Structure of an Agent-based IED

The agent-based IED's structure is shown in fig. 3. Agents within an IED perceive their environment through local sensors and act upon it through the IED's actuators. Examples of sensor inputs might include local measurements of current, voltage, and breaker status. Actuator outputs might include signals to initiate breaker trips, transformer tap setting adjustments, and capacitor bank selection. Agents might even interface with legacy systems such as Supervisory Control and Data Acquisition (SCADA) systems. The host computer shown in fig. 2 could act as a bridge between the old and new systems in this situation. Internally, agents might be composed of many layers of functionality and control or may be contained in a single layer depending on the designer's specifications. As shown in fig. 2, agents have the ability to communicate through a LAN in order to interact with other agents directly located on that same LAN, or can pass information along to a WAN, such as the Utility Intranet, ultimately communicating with more remote IEDs.

The increasing use of agents in IEDs makes an agent-based framework a natural choice. Protection and control engineers can create agents for use in real situations and test them with minor modification in the EPOCHS environment. Agents can also mimic the behavior of traditional systems. EPOCHS' early adopters have found the agent concept to be intuitive.

C. The Structure of a Utility Communication Network

Networked computing systems have become ubiquitous, and will likely become more widespread within electric utility systems. Technology is constantly changing, but it is possible to make educated guesses about what utility communication systems will look like. First, the network systems will almost certainly be built from standard commercial off-the-shelf components. To do otherwise would be expensive both in initial cost outlay and in system maintenance. Thus, these networks will be based on Internet standards even if the systems remained independent of the global network conglomeration. Signs that such changes are coming are already appearing in efforts such as the TCP-based UCA [28].

D. The Agent's API in EPOCHS

EPOCHS' agent API is intended to minimize the differences between simulated systems and their real-world counterparts, as well as to ease implementation for EPOCHS' users. The basic functionality is shown in fig. 4. Functions have been broken down into two main categories. Events occur in the AgentHQ subsystem and notification is passed on to the appropriate agents. Agents interact with their surroundings by using method calls to get and set the state of their environment and to exchange messages with other agents.

```
Interface Agent
{
methods:
   double get_round_time();
   void send_comm_msg(comm_type, group, src, dst,
                pkt_size, msg);
   void send_power_msg();
events:
   void request();
   void action();
   void recv_comm_msg(comm_type, group, src, dst,
                pkt_size, send_time, round_num, msg);
   void recv_power_msg(msg);
};
```

Fig. 4. The Agent Interface

AgentHQ is triggered at each synchronization point and acts as a proxy between the agents, the network communications simulator, and the active electric power simulator. At that time, the AgentHQ calls each of the agent's request and action methods giving them an opportunity to calculate their set of operations for the next time step.

The agents remain dormant until one of their event methods is called. Power system agents mimic those in real protection and control systems by polling their environment at regular intervals. When the beginning of an interval is reached, each agent is given a chance to request its power system state through the send_power_msg method and receive the results through the recv_power_msg event notification. All agents' initial requests are sent and the replies are received in one block. This is an optimization to help compensate for the use of files to exchange information between the AgentHQ and the electric power simulation systems. When all agents have

been given a chance to run, the AgentHQ allows each to react to their current state in the action method where they can send communication messages using the send_comm_msg method, or can make additional power system get and set requests using the send_power_msg method. In addition to these regular activation intervals, individual agents may receive communication messages through the recv_comm_msg event at any time and can take additional actions in their response.

VI. Experimental Results

This section describes a case in detail where EPOCHS has been used to demonstrate the benefits and drawbacks in using communication in an agent-based special protection system. A brief overview of a backup protection system, using PSCAD/EMTDC and NS2, is also provided. An additional test case for EPOCHS, using PSLF, and NS2, monitors power systems to prevent blackouts caused by voltage collapse. Full details of all cases can be found in [31]. These experiments point towards a future where tests can be performed to find problems before systems are deployed.

It must be emphasized that EPOCHS allows user to test either electric power protection and control schemes using the PSCAD/EMTDC electromagnetic simulator in conjunction with the NS2 network simulator or in electromechanical simulations using PSLF in conjunction with NS2. EPOCHS has never been used with PSLF, PSCAD/EMTDC, and NS2 simultaneously. The vastly different timescales involved in electromagnetic and electromechanical simulations make their combination difficult if not impossible to model properly.

A. Backup Protection Case

Relay misoperations, such as incorrectly tripping a healthy line or the failure to isolate a faulted line, are involved in major disturbances in the power system [32]. Zone 3 backup relays are required to clear a fault when the primary relay fails. Backup protection systems have many challenges to overcome. First, the region they isolate is often larger than it need be. Second, they traditionally act without the use of explicit communication. The need for small isolated regions imposes long lag times for zone 3 relays, which are one of the major causes of power system instability. This section outlines a backup protection system that improves on traditional systems using explicit network-based communication.

A new agent-based design for zone 3 backup protection relays has been created to improve on traditional relays. The agent-based relays are able to use communication to send their relay status information, breaker trip signal events, and local measurements when a primary protection event occurs.

The agent-based protection system has many benefits over traditional systems. Zone 3 relays can be monitored to allow corrections to prevent false breaker trips. These corrections have the potential to greatly reduce the number of incorrect trips in cases where there is a heavy load. In addition, in the case of both primary and local backup relay failure, the agent can locate the faulted line using notification messages and can send a trip signal to potentially only clear the faulted line. Traditional backup systems clear such faults using remote backup relays with a bigger isolated region and with a greater delay time. If an incorrect primary relay trip is found by the agent then a block signal can be sent to stop an unwarranted breaker trip. Breaker failure protection trips all breakers connected in the same bus in most bus arrangements and induces a big disturbance leading to poor overall reliability. The reliability gains can be significant in those cases. As an example, fig. 5 shows a 400 kV power system. Fig. 6 shows the result of a primary relay misoperation. Traditional relays do not explicitly communicate and fig. 6 (a) and (b) show that the primary relay misoperation occurs when they are used. Agents are able to coordinate their actions to detect and stop an incorrect breaker signal, as shown in fig. 6 (c) and (d).

The backup protection example demonstrates EPOCHS' utility in simulating electromagnetic cases that depend on communication using PSCAD/EMTDC with NS2. Fig. 6 illustrates the potential benefit of coordinating backup protection through communication. The SPS example in the next section is an electromechanical case using PSLF with NS2. Full details of these cases can be found in [31].

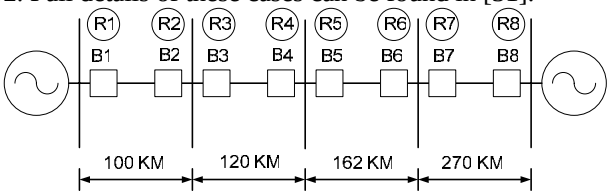


Fig. 5. A 400 kV power system with an agent-based backup protection system.

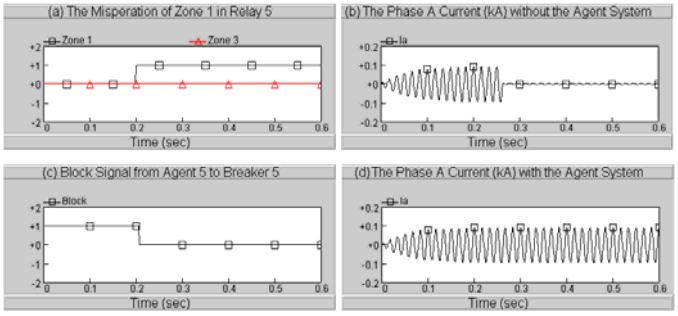


Fig. 6: (a) A primary relay misoperates at time 0.2 by setting a trip signal. (b) A traditional backup protection system would allow the breaker to trip, cutting power across the line. (c) The agent-based protection system communicates with its neighbors and determines that the trip signal should be blocked. (d) The agent-based system blocks the false trip signal.

B. SPS Case Overview

Power system instability usually involves large areas and results in dire consequences. Loss of generator synchronism for a single or group of generators with respect to another group of generators is a transient instability, which can result in costly power blackouts. Stability problems typically involve disturbances such as the loss of generation, loads, or tie lines. These disturbances stimulate power system electromechanical dynamics. The system response typically includes deviations in frequencies, voltages, and generator phase angles. Special Protection Schemes (SPS) are mechanisms generally designed to counteract power system instability. The most common SPS schemes employ generation rejection and load shedding, according to a report on SPSs throughout the world [33].

The SPS in this section is designed to react to a severe fault in a major EHV transmission line in the case where another outage has already taken place in the same corridor. The goal

is to prevent instability and preserve the system's integrity within a safe operating frequency range. The SPS system employs an algorithm to determine the proper amount of load to shed in order to hold the system's frequency above a preset level based on wide area measurements. This SPS is designed for wide-area protection and acts in a system-oriented manner. The SPS requires synchronized information periodically sampled across the system and is heavily dependent on an underlying communication infrastructure. The wide-area protection system's reliance on communication requires a simulation system that incorporates detailed models of both network communication and electric power systems. EPOCHS is the only platform that provides these combined capabilities. The proposed SPS system has been tested with a modified version of the IEEE 50 generator test case. The results show promise for the use of SPS schemes like the one described here in the future. The SPS experiments also demonstrate the utility of EPOCHS in evaluating electromechanical systems that make use of network communication.

C. An Algorithm to Estimate a System's Disturbance Size

This section describes an algorithm, which estimates a disturbance's size when confronted with electromechanical dynamics during an emergency. The method allows one to compute the steady state frequency after a disturbance and the amount of load shedding required to maintain a preset frequency level taking the effect of generator governors into account. Experiments described in section C and results presented in section D demonstrate the viability of this technique in a realistic test environment within EPOCHS.

The goal of the proposed SPS is to shed enough load to keep the system's frequency above a preset level following a generation loss. The key to doing so is to determine the precise generation shortfall when a disturbance occurs. The system shortfall can be calculated with the following formula.

$$P_d = P_a + \Delta P_e\left(\omega_{0^+} - \omega_{0^-}, u_{0^+} - u_{0^-}\right) \quad (1)$$

Formula 1 shows that the size of the disturbance, P_d, is equal to the system accelerating power, P_a, which is proportionate to the change in the system's frequency, plus the change in electrical power demand ΔP_e due to the variation in frequency and voltage. P_d is the key to determining the amount of generation that has been lost. It is important to note that $_{O^-}$ and $_{O^+}$ respectively denote the time immediately before and after the disturbance. P_a and ΔP_e can both be obtained based on wide area measurements using the generators' operating status and samples of the system's frequency before and after the disturbance, but measurements must be simultaneously taken at points throughout the region. Data points must be sent to the SPS from the region's generators and key loads, and action must be taken, within a half second to be effective. The expense and requirements involved in deploying the SPS scheme make it critical that the system is thoroughly tested so that its operations are fully understood before deployment. These factors make the SPS system an excellent candidate for evaluation in EPOCHS.

D. The System Studied and Agent-Based SPS Scheme

1) The System Studied

Experimental simulations make use of IEEE's 145-bus 50-generator system [34]. The 50-generator test case has a large share of the generation concentrated in the northeast and high load in the southwest of the system.

The IEEE 145-bus 50-generator test case has been modified so that it is more representative of power systems that require SPS protection. The six machines located at buses 93, 104, 105, 106, 110, and 111 are represented with two-axis machine models equipped with IEEE type AC4 exciters. The other generators are represented by classical machine models. All generators are equipped with basic steam turbines and employ governors with a 5% droop setting. System analysis has been performed after the governors have responded, but before new load reference set points are established by the AGC subsystem. The system has been modified by adding a 500 kV line from bus 1 to bus 25. The added line increased the number of system branches from 453 to 454. The intent is to create a case that requires the use of a SPS in order maintain system stability. Power systems can generally sustain the loss of a single tie line, but require action with the loss of a second if the line is not cleared quickly enough. Next, the total system capacity has been reduced to 30,050.00 MW. This lower system capacity makes the 4,277 MW power flow along the main 500 kV transmission corridor much more important than it is in the original system. The changes mentioned above caused the admittance load to be abnormally high. The 145-bus IEEE system has been rebalanced to correct the problem by setting the percentage of admittance load to 5.02%. The remaining 94.98% of the system's load has been set as constant active and reactive power.

2) Description of the SPS Architecture

The proposed SPS is required to act rapidly and reliably to electromechanical instability. The communication requirements are different from that of traditional SCADA systems due to the need for fast information updates and rapid response to commands dictating generation rejection and load shedding. The agents are separated into three types: the main SPS agent, generator agents, and load agents. In the IEEE 50-generator example, the main SPS agent is located at bus 1, a 500 kV substation. The main agent identifies extreme contingencies such as the loss of two lines and performs both generation rejection with preset units and load shedding based on real-time measurements. The generators that can be rejected are selected based on simulation studies. The main SPS agent communicates with generation and load agents to gather data values including generators' connection status, active power outputs, angular frequencies and frequency derivatives. It also communicates with agents located at major system or load buses to collect voltage and frequency measurements as well as the load available for shedding.

Generator agents are located at power plants where they send their measurements to the main SPS agent upon request.

Generator agents also reject generation when ordered to do so by the main SPS agent. Load agents are mainly located at distribution substations. These agents shed load when ordered to do so by the main SPS agent. They also locally perform underfrequency load shedding (UFLS). UFLS might occur if the frequency reached a threshold value of 57~58.5Hz after a remote load shedding scheme with a preset frequency of 58.8 Hz failed to hold the frequency above 58.5 Hz.

E. Simulation Results

Initially in the electrical portion of the EPOCHS-based simulation, the 500 kV transmission lines from buses 1 to 6, 1 to 25, and the two lines from 1 to 2 carry respectively 722.2 MW, 1,106.5 MW, and 2×361.1 MW. A trip on the 1-6 line causes the system to operate under stressed conditions. The power in lines 1-25 and 1-2 respectively increase to 1,384MW and 2×515.7 MW. If a three phase fault occurs near bus 1 when line 1-6 has already tripped then the critical fault clearing time to open line 1-25 is around 0.065 seconds.

A communication network was created with a node at each bus and an edge between nodes wherever one or more transmission lines existed. The link delays were set to 0.5 seconds. Each link's bandwidth was set to 150 Megabits per second so that no packets would be lost due to network congestion. Per-link loss rates were first set to 0% and then the experiment was rerun with a loss rate of 5%. The simulation with 5% link loss rates was performed to get a basic idea of how the SPS system would react to lossy communication. All messages were sent using UDP.

The assumptions made in the communication network configuration helped to validate that the SPS system was operating properly and that the results obtained from EPOCHS were correct. Actual network traffic is delayed due to propagation delays, router queuing delays, and buffering inside the sending and receiving machines. Queuing delays have an inherent dependency on the network congestion levels created by competing traffic, which tends to vary over time. EPOCHS has the ability to model these complexities, but early experiments have modeled traffic simply in order to aid in the validation of EPOCHS and the SPS system studied.

In the SPS experiments, a fault occurs on the 1-25 line at time 0. The fault is cleared at 0.07 seconds, and a trip command is sent to generator 93 at 0.10 seconds. Because the fault is cleared after the critical fault clearing time, the system becomes transiently unstable and one group of 17 generators loses synchronism with another group of 33 generators. Fig. 7 shows the rotor angle of a representative sample of generators with respect to the generator at slack bus 145.

The main SPS agent at bus 1 recognizes the situation and begins to communicate with other system agents to gather data values including generators' connection status, active power outputs, and angular. The main agent also issues a generation rejection order to the agent at bus 93. Bus 93 is chosen based on an off-line simulation study. Generation rejection keeps the system stable, as shown in fig. 8 (a), but the frequency drops near 57.5 Hz, as shown in fig. 8 (b).

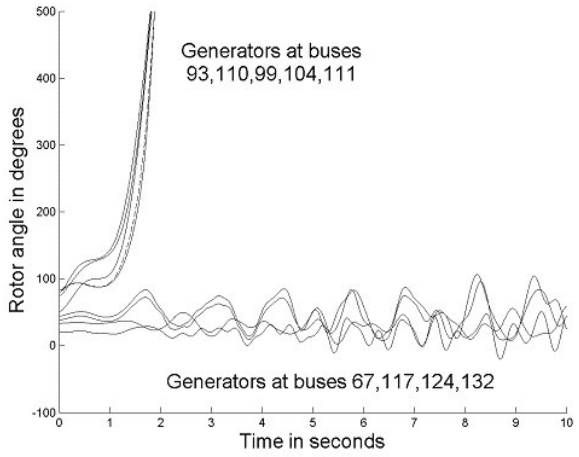
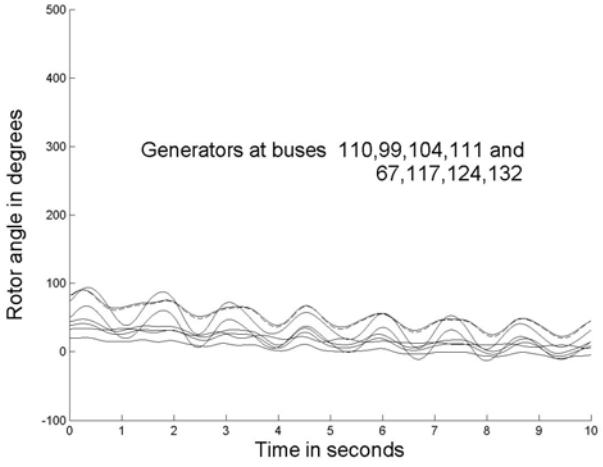
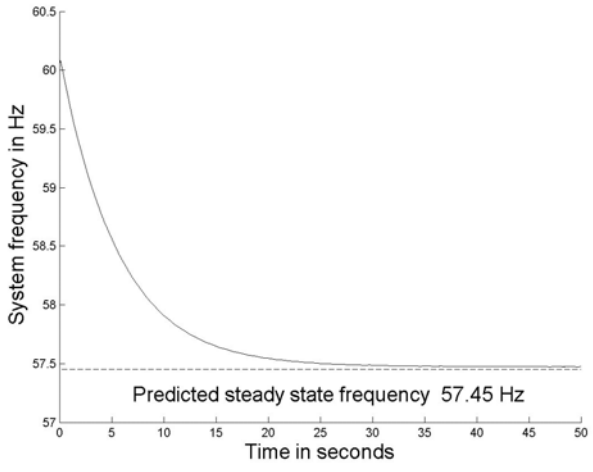


Fig. 7: Rotor angles with transient instability induced by a disturbance



(a)



(b)

Fig. 8: (a) The system remains stable after generation rejection. (b) The system undergoes an unacceptable frequency decrease.

The main agent detects the "disturbance" created by generation rejection at bus 93. It begins to estimate the disturbance size and finds it to be -1,862 MW. The main agent calculates that there is 2,090 MW generation remaining and predicts that the steady state system frequency after this

disturbance is 57.45 Hz. The predicted frequency value, as shown as in fig. 8 (b), is very close to the simulation results.

Because the preset frequency is 58.8 Hz in the test system, remote load shedding is required. The load shedding amount required to maintain the system frequency above that preset level is estimated to be 886 MW. This total load is shed at buses 14, 25, 27, 63, and 69 with the same percentage allocated to each load. The shed loads are, respectively, 85.97 MW, 293.29 MW, 182.25 MW, 157.16 MW and 167.92 MW. The system's frequency response, as shown in fig. 9, demonstrates the merit of the proposed SPS system.

The SPS experiments also validated the accuracy of EPOCHS. Logs of agent's network communication, electric power requests, and communication sent between EPOCHS' components show that EPOCHS was operating correctly. Experiments operated as predicted, which also validated the EPOCHS simulation environment. EPOCHS was also helpful to the system developers in illustrating potential problems that had not been anticipated in the SPS system.

Fig. 10 shows that the SPS operation with a 5% loss rate per link is different from the one without communication losses. The algorithm used by the SPS agent to determine the amount of load to shed depends on measurements that are simultaneously taken at each of the load and generation agents in the region. Measurements must be taken close to the point of the disturbance to be valid. The likely reason for the differences between the 0% loss rate and 5% loss rate cases is that the information used to calculate the disturbance size and load shedding amount are different for the two cases. If too many communication packets are lost then there can be a large time lag between the beginning of the disturbance and the point where that disturbance was computed. The system frequency will continue to decline after the disturbance has occurred. The SPS algorithm implicitly depends on receiving measurements close to the time of the fault, but no mechanism has been explicitly created to ensure this in the experiments described. The results shown in fig. 9 serve as an illustration of the utility of a simulation platform like EPOCHS, and also demonstrate the negative consequences that hidden flaws can have if realistic simulation facilities are not available.

The experimental results demonstrate the utility of the proposed SPS system. More generally, the experiments validate the EPOCHS platform and illustrate its ability to effectively simulate scenarios involving power protection and control systems based on network communication.

VII. Future Work

EPOCHS and its applications open many possibilities for future work. First, large applications need to be developed to model the complex interactions that take place in real power systems. Second, it is likely that the utility intranets used to support next-generation protection and control systems will be shared between a variety of devices generating a wide range of network traffic. Models of the types of background traffic that might be expected in these intranets have been developed [31] and are being integrated into EPOCHS. Finally, a goal of the project team is to use EPOCHS to model a real power disruption, such as the August 2003 blackout.

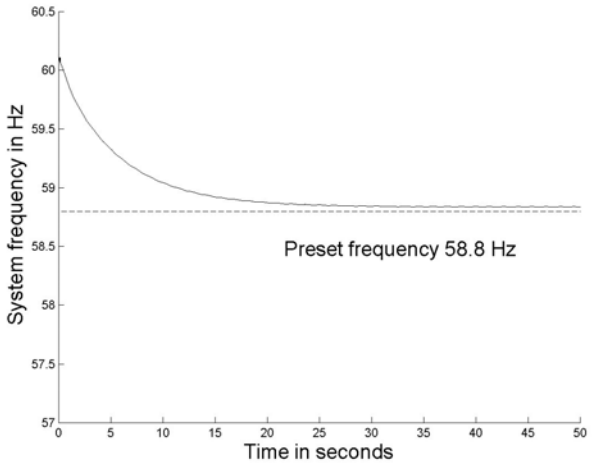


Fig. 9: Remote load shedding maintains the frequency above a preset level

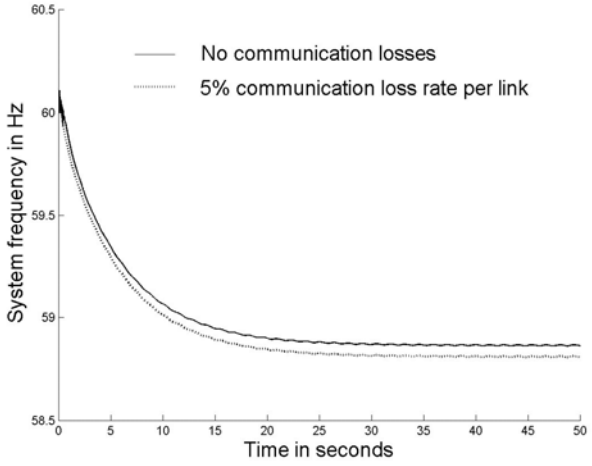


Fig. 10: Comparison of the SPS operation with different levels of loss rate

VIII. Conclusion

This article has presented EPOCHS, a simulation engine that combines PSCAD/EMTDC, PSLF, and NS2 functionalities together with an agent component. The paper has three main contributions.

- The simulator is the first to combine realistic network communications with electric power components.
- EPOCHS illustrates methods for bridging unrelated simulation engines without making use of source-code modification to any of the systems in question.
- The EPOCHS platform uses a simple yet powerful agent framework, which is easy to use for simulation modelers.

EPOCHS' utility has been demonstrated in electric power scenarios involving communication including backup protection, special protection systems, and voltage collapse.

References

[1] F. Kuhl, R. Weatherly, and J. Dahmann, *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Upper Saddle River, NJ: Prentice Hall, 1999.
[2] S. J. E. Taylor, R. Sudra, T. Janahan, G. Tan, and J. Ladbrook, "Towards COTS Distributed Simulation Using GRIDS," presented at Proceedings of the Winter Simulation Conference, Piscataway, New Jersey, 2001.
[3] M. Tucci and R. Revetria, "Different Approaches in Making Simulation Languages Compliant with HLA Specifications," presented at Proceedings of the Summer Computer Simulation Conference, 2001.
[4] S. Strassburger, "On the HLA-based Coupling of Simulation Tools," presented at European Simulation Multiconference, 1999.
[5] C. McLean and F. Riddick, "The IMS Mission Architecture for Distributed Manufacturing Simulation," presented at Proceedings of the Winter Simulation Conference, Piscataway, New Jersey, 2000.
[6] Development Coordination Group of EMTP, "EMTP-RV," http://www.hydroquebec.com/emtp/software/emtp_rv.html, Accessed June 10, 2005.
[7] ATP CanAm Users Group, "ATP," http://www.ee.mtu.edu/atp/, Accessed June 10, 2005.
[8] Siemens AG, "NETOMAC," http://www.netomac.com/pubdown.htm, Accessed June 10, 2005.
[9] OPNET Technologies Inc., *Modeling Concepts Reference Manual Product Release 10.5*. Bathesda, MD: OPNET Technologies, Inc., 2004.
[10] Scalabe Network Technologies, "QualNet," http://www.scalable-networks.com/, Accessed June 10, 2005.
[11] RTDS, "RTDS," http://www.rtds.com, Accessed June 10, 2005.
[12] T. Technologies, "HYPERSIM," http://www.transenergie-tech.com/en/produits/hypersim.html, Accessed June 10, 2005.
[13] The Mathworks Inc., *Getting Started with Matlab Version 7.0*, Sixth ed. Natick, Massachusetts: The Mathworks Inc., 2004.
[14] RDSoft, "ARENE," http://rdsoft.edf.fr/power_systems_simulation.html, Accessed June 10, 2005.
[15] L. F. Wilson, D. Burroughs, J. Sucharitaves, and A. Kumar, "An Agent-based Framework for Linking Distributed Simulations," presented at Proceedings of the Winter Simulation Conference, Piscataway, New Jersey, 2000.
[16] S. Lee, A. Pritchett, and D. Goldman, "Hybrid Agent-based Simulation for Analyzing the National Airspace System," presented at Proceedings of the Winter Simulation Conference, Piscataway, New Jersey, 2001.
[17] M. Yalla, M. Adamiak, A. Apostolov, J. Beatty, S. Borlase, J. Burger, S. Dickson, G. Gresco, W. Hartman, J. Hohn, D. Holstein, A. Kazemi, G. Michael, C. Sufana, J. Tengdin, M. Thompson, and E. Udren, "Application of Peer-to-Peer Communication for Protective Relaying," *IEEE Transactions on Power Delivery*, vol. 17, pp. 446-451, 2002.
[18] H. Doi, Y. Serizawa, H. Tode, and H. Ikeda, "Simulation Study of QoS Guaranteed ATM Transmission for Future Power System Communication," *IEEE Transactions on Power Delivery*, vol. 14, pp. 342-348, 1999.
[19] C. Taylor, D. C. Erickson, K. E. Martin, R. E. Wilson, and V. Venkatasubramanian, "WACS - Wide-Area Stability and Voltage Control Systems: R&D and Online Demonstration," *Proceedings of the IEEE*, vol. 93, pp. 892-906, 2005.
[20] M. Baran, R. Sreenath, and N. R. Mahajan, "Extending EMTDC/PSCAD for Simulating Agent-Based Distributed Applications," *IEEE Power Engineering Review*, pp. 52-54, December 2002.
[21] IEEE, "HLA Framework and Rules," 1516-2000, 2000.
[22] S. A. Colby and D. L. Beetham, "Long Range Artillery Simulation Using Component Based Development Techniques and the High Level Architecture," presented at Winter Simulation Conference, Orlando, Florida, 2000.
[23] E. Gottlieb, M. J. McDonald, F. J. Opped, J. B. Rigdon, and P. G. Xavier, "The Umbra Simulation Framework as Applied to Building HLA Federates," presented at Winter Simulation Conference, San Diego, California, 2002.
[24] H. Hibino, Y. Fukuda, Y. Yura, K. Mitsuyuki, and K. Kaneda, "Manufacturing Adapter of Distributed Simulation Systems Using HLA," presented at Winter Simulation Conference, San Diego, California, 2002.
[25] Manitoba HVDC Research Centre, *PSCAD/EMTDC Manual Getting Started*. Winnipeg, Manitoba, Canada, 1998.
[26] General Electric, "PSLF Manual," vol. 2003, 2003.
[27] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidermann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in Network Simulation," *IEEE Computer*, vol. 33, pp. 59-67, 2000.
[28] IEEE, "UCA Version 2.0," TR 1550, November 1999.
[29] R. M. Fujimoto, *Parallel and Distributed Simulation Systems*. New York, NY: Wiley-Interscience, 2000.
[30] IEC, "61850," 2002-2004.
[31] K. M. Hopkinson, "Overcoming Communication, Distributed Systems, and Simulation Challenges: A Case Study Involving the Protection and Control of the Electric Power Grid Using a Utility Intranet Based on Internet Technology," Doctoral Dissertation, Ithaca, New York: Cornell University, 2004, pp. 245.
[32] A. G. Phadke and J. S. Thorp, "Expose Hidden Failures to Prevent Cascading Outages," *IEEE Computer Applications in Power*, pp. 20-24, 1996.
[33] P. M. Anderson and B. K. LeReverend, "Industry Experience with Special Protection Schemes," *IEEE Transactions on Power Systems*, vol. 11, pp. 1166-1179, 1996.
[34] IEEE Committee, "Transient Stability Test System for Direct Stability Methods," *IEEE Transactions on Power Systems*, vol. 1, 1992.

Kenneth Hopkinson received his B.S. in Computer Science from Rensselaer Polytechnic Institute in 1997, and his M.S. and Ph.D. in Computer Science from Cornell University in 2002 and 2004, respectively. He is an Assistant Professor of Computer Science at the Air Force Institute of Technology. His research interests include distributed systems, networking, and simulation.

Xiaoru Wang received a B.Sc. in Electrical Engineering and a M.Sc. in 1983 and 1988 respectively, from Chongqing University, China, and a Ph.D. from Southwest Jiaotong University, China, in 1998. She is a Professor in the School of Electrical Engineering, Southwest Jiaotong University, China. Her research interests lie in the area of power system protection and control

Renan Giovanini received a B.Sc. in Electrical Engineering and an M.Sc. in 1998 and 2000, respectively, from the EESC – University of São Paulo at São Carlos in Sao Carlos, Brazil, where he is completing his Ph.D. His main research interests are power systems protection and artificial intelligence.

James Thorp is the chair of the Department of Electrical and Computer Engineering at Virginia Polytechnic Institute. He was an associate editor for IEEE Transactions on Circuits and Systems from 1985 to 1987. He is a member of the National Academy of Engineering, a Fellow of the IEEE, and a member of the IEEE Power System Relaying Committee, CIGRE, Eta Kappa Nu, Tau Beta Pi, and Sigma Xi.

Kenneth Birman is a Professor of Computer Science at Cornell University. A Fellow of the ACM, Birman has published extensively on reliable, secure distributed computing since joining Cornell in 1982. He developed the Isis Toolkit, which controls communications in the New York Stock and Swiss Exchanges, the French air traffic control system, and oversaw the development of the Horus, Ensemble, and Spinglass systems. He was Editor in Chief of the ACM Transactions on Computer Systems from 1994-1999.

Denis Coury received a B.Sc. degree in Electrical Engineering from the Federal University of Uberlandia, Brazil in 1983, a M.Sc. from the University of São Paulo, Brazil in 1986, and a Ph.D. from Bath University, England in 1992. He is a Professor in the Department of Electrical Engineering, in the University of São Paulo at São Carlos, Brazil. His research centers on power system protection and control, expert systems, and neural networks.