ISGT 2011: IEEE PES Conference on Innovative Smart Grid Technologies, January 17-19, 2011, Anaheim, CA



# Power System and Communication Network Co-Simulation for Smart Grid Applications

Hua Lin, Santhoshkumar Sambamoorthy, Sandeep Shukla, James Thorp, Lamine Mili

Department of Electrical and Computer Engineering, Virginia Tech

***Abstract*—The vision of a smart grid is predicated upon pervasive use of modern digital communication techniques to today's power system. As wide area measurements and control techniques are being developed and deployed for a more resilient power system, the role of communication network is becoming prominent. Power system dynamics gets influenced by the communication delays in the network. Therefore, extensive integration of power system and its communication infrastructure mandates that the two systems are studied as a single distributed cyber-physical system. This paper proposes a power/network co-simulation framework which integrates power system dynamic simulator and network simulator together using an accurate synchronization mechanism. The accuracy is tunable based on the time-scale requirements of the phenomena being studied. This co-simulation can improve the practical investigation of smart grid and evaluate wide area measurement and control schemes. As a case study an agent-based remote backup relay system is simulated and validated on this co-simulation framework.**

***Index Terms*—Co-Simulation, Remote Backup Relay, Smart Grid**

## I. Introduction

In December 2008, a report was prepared by the participants of a workshop organized by the department of homeland security [1]. This report suggested the need for a national power grid simulator, and recommended various desirable aspects of such a simulator. Such a simulator should allow for modeling various possible disruptive events, studying interdependencies between the power grid and other critical infrastructures, and allow for planning and design of smarter capabilities of the national grid to enhance resilience, robustness, integration of renewable energy sources. Even though this report does not specifically deal with the embedded computational and communication capabilities envisioned for the future smart grid, there are already many efforts worldwide to enable various power grids to communicate data in real-time over wide areas, and use networked and distributed control to avoid various disastrous scenarios including blackouts, unwarranted generation shutdowns, unwarranted frequency excursions, inter-area oscillations, voltage instability, and so on.

Various commercial companies and universities have developed a number of tools over the decades to model, analyze and understand the power system dynamics in scalable and computational ways [2, 3]. These tools study the continuous dynamics of a continuous state, continuous time systems usually described by ordinary differential equations (ODE) or partial differential equations (PDE). Numerical methods to compute approximate trajectories of such systems are quite matured. Depending on the time-scales at which interesting state changes happen, the discretization of time and integrating the dynamics along the time line are carried out. Similarly, simulation techniques for discrete event and discrete time systems are also well developed over the years. Data networking such as the Internet, and other communication networks have been simulated using discrete event simulation techniques, because such systems are inherently discrete, and hence time driven simulation would induce unnecessary inefficiency.

The smart grid vision suggests that the power grid will have extensive networking to communicate various measurements and remotely affect control. Some of these have already started with the Phasor Measurement Units (PMUs) based phasor data collection, and communication of phasor values (at 30 times a second rate) to Phasor Data Concentrators (PDC) for real-time state estimation and various wide-area control of the power systems. However, the structure of the communication network to be laid out in the national power grid, the communication protocols to be used, the physical media, the distributed algorithms to make decisions on power system state and required control actions, the hierarchy of communication and control network, and many other issues remain unsettled to date. This mandates that we need power system and communication network co-simulation as opposed to only a national power grid simulator as suggested in the report [1].

If we can implement an effective, scalable and efficient power grid and communication network co-simulator, we can design wide-area measurement and control schemes that have not been imagined yet, and easily simulate its effect, optimize the design and cost, choose the right communication protocols in order to fit the communication delays within the time-scales of the power system events that would be affected by such communications. However, it is not an easy task to get this done, as there is a mismatch in the models of computation in the two simulation worlds. Discrete event and continuous time dynamics have to interact together.

In the past two well known approaches have been reported on such co-simulation. The EPOCHS [4] approach and the ADEVS [5] approach. The EPOCHS approach is based on federated simulation with multiple simulation tools, some of which are discrete-event, and some are continuous time. The interfacing is achieved through a mediator software component, which synchronizes the multiple simulators by allowing them to exchange data periodically. As we show in Section II in this paper, this approach not only poses extra overhead of an intermediary of synchronization, it actually can

lead to a mismatch between the real dynamics and interaction, and the simulated one. We alleviate this problem by making an asymmetric approach where the discrete event simulator considers the integration time steps of the simulator of the continuous dynamics, as events to be scheduled. It can also adaptively adjust the granularity of its own event spacing. In a certain sense, our approach is closer to the EPOCHS as we also believe in federation of existing simulation frameworks as opposed to inventing our own simulators. That is a much more proficient way to achieve the co-simulation as the existing simulators are developed over multiple decades, and effectively used in the industry. The ADEVS approach does not federate. It actually requires its own modeling of the continuous dynamics of a power system in the DEVS framework. The DEVS approach is to convert the continuous dynamics into discrete events by different state event detection mechanisms (e.g. zero crossing). Since most commercial power system modeling and simulation tools do not use this approach, the federated approach will not readily apply to the ADEVS approach. We believe that our approach improves the EPOCHS approach in accuracy while keeping intact the idea of federation and our accuracy should be similar to ADEVS approach while improving upon it by allowing faster construction of co-simulation. In this paper we not only describe our approach in detail, we show using a case study that our approach is effective, and it is much easier to set up the co-simulation and obtain results.

The following paper is organized as: Section II presents the co-simulation framework with an accurate synchronization mechanism between continuous time and discrete event simulation. Section III describes the co-simulation construction using a transient power simulator and a network simulator. An agent-based remote backup relay scheme which benefits from communication infrastructure is studied and discussed in Section IV. This scheme is further simulated using our co-simulation tool in Section V to demonstrate the effectiveness of our method. In the end the whole paper is concluded in Section VI.

## II. Co-Simulation Framework

Power system dynamic simulation is commonly modeled as a continuous time simulation. A typical case is transient stability simulation. Communication networks are usually modeled as discrete events systems to account for the stochastic nature of packet generation and transmissions. Hybrid models [6-8] integrating those two have been applied in the area of computer engineering which aimed at mixed signal circuit designs but still rarely seen for power system except the ADEVS [5] approach.

### *A. Continuous System Simulation*

In a continuous time system, the system state variables change in a continuous manner with respect to time. Typically the system is represented by sets of differential equations in which the relationship between continuous state variables' changing rates are defined. For simple cases, the differential equations can be solved analytically to get closed form results. However for most cases in the real world such closed form results are not available. The solution for this is to numerically solve the problem by discretizing differential equations and time, then integrating small variations of state variables to approximate the system trajectory. The discretized time step is often very small so that the system variables do not have an abrupt transition within the time step. The discretization and integration process can be represented as:

$S$ : System State Variables — $S_n$ : Sample of S at $t_n$

$X$ : System Input Variables — $F$ : Discretized Equations

$Y$ : System Output Variables — $G$ : Output Functions

$$S(t_0) = S_0;$$
$$\Delta S_n = F(S_n, X_n, \Delta t_n);$$
$$S_{n+1} = S_n + \Delta S_n;$$
$$Y_n = G(S_n, X_n, \Delta t_n);$$

Especially in power system analysis, dynamic simulation runs in a step by step manner which can be seen in Fig. 1. After the system initialization by solving system power flow, the simulation enters a loop. Within this loop, first the network boundary variables for dynamic models connected directly to the system network are calculated. For example, the generator internal voltage is calculated from source current. Then dynamic models' secondary variables are calculated from system state variables. This includes models that are not directly connected to the system. For example some internal control devices such as excitation systems and governors are processed in this step. One round in the loop is completed by calculating state variable derivatives and integrations. The loop continues until the simulation reaches a preset stop time.

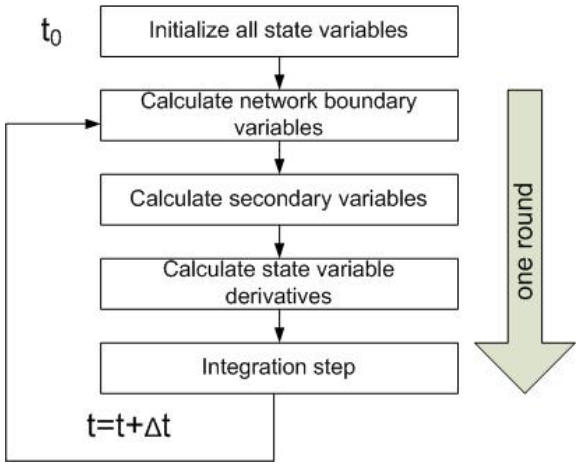


Figure. 1. Step by step simulation of power system dynamics

### *B. Discrete Event Simulation*

Discrete event-driven simulation is suitable for systems whose state is subject to change due to discrete events. The occurrences of events are usually unevenly distributed on a timeline. For instance, the arrival time of cars to a toll booth may follow Poisson distribution. Hence time discretization into steady small time intervals as done in continuous time systems cannot be appropriately applied to discrete event system since the time step is difficult to determine. If the time step is selected too small then it will waste simulation times since system state remains unchanged during many

consecutive time steps. If the time step is selected too long, then many events could be missed during a single time step.

In an event-driven simulation, instead of doing integration of discretized time, the time hops between events. A event scheduler is designed to record current system time and also maintain an event list. Event list is a queue that stores system events with timestamps in chronological sequence. Before a simulation starts the scheduler initializes the system state and the event list. When the simulation starts, the scheduler proceeds with the first event from the list and sorts out the processes attached to this event. The processes can generate new events to and remove existing events from the event list. New events are placed to the right places in the list by the scheduler. Then the scheduler switches the system time directly to the time when next event is going to happen. The whole simulation stops when system time reaches a certain stop time or the system reaches a certain state.

Communication networks simulation is an example that uses discrete event-driven simulation. The system consists of network hardware devices as well as software agents. The events in communication network could be receiving or sending a packet as shown in Fig. 2. The network protocols receive packets and send new packets to the network.

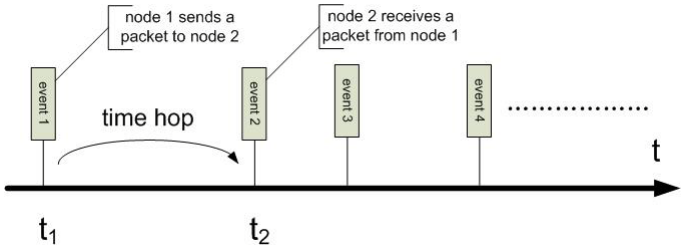


Figure. 2. Event-driven simulation for communication network

### C. Co-Simulation Framework

Under the smart grid scenario, the power system interacts with its communication network infrastructure at widely distributed locations. Software agents collect power system data through Advanced Metering Infrastructure (AMI) and send them to corresponding receivers through communication network. Analyzing the systems together is a natural outcome and hybrid simulation tools combining power system and communication network simulation are needed to cover new design areas. A precise synchronization mechanism is the key to fulfill this requirement.

When researchers first made efforts to integrate power system and communication network simulation an intuitive synchronization is adopted as shown in Fig. 3 [4]. In this mechanism power system simulation and communication network simulation runs independently until both simulators' simulation time reaches a synchronization point, as denoted in Fig.3 by red dashed lines. At this point both simulators halt themselves and the overall control is taken over by an external mediator which has access to internal data of both simulators. The mediator imitates the behavior of agents in the smart grid, exchanges data between two simulations and executes necessary modifications to them. The control will be returned to two simulators after the coordination and the two simulators will continue their independent simulations until the next synchronization point. If the power system simulation time step is steady then there will be equal number of simulation rounds between synchronization points. However there can be any number of network events that happene between synchronization points due to their uneven distribution.

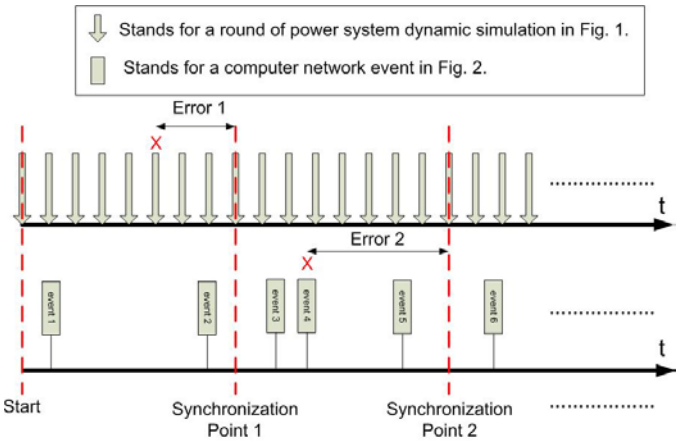


Figure. 3. Synchronization mechanism with accumulating errors

This synchronization mechanism can introduce time accumulating errors owing to difficulty for selecting a perfect synchronization boundary. For instance, between two synchronization points if there is a fault that happened in the power system, the simulator has to store its fault report temporarily and wait to the synchronization point to pass the report to mediator and components in network simulator. This is indicated by "Error 1" in Fig. 3. On the other hand, if a control message is received by an agent in network simulator, it as well has to wait until synchronization to take action in the power system. This is also presented by "Error 2" in Fig. 3.

We propose a new co-simulation framework which removes the accumulating errors described in the preceding paragraphs. Since the disadvantage of that synchronization is very similar to the one that using continuous time simulation methods to simulate a discrete event system, we run the simulation globally in a discrete event-driven manner as shown in Fig. 4. In this framework we remove those explicit synchronization points and make the simulators implicitly synchronized. We adopt a global scheduler for co-simulation and two simulators share the same timeline instead of running independently. The repeating rounds in power system dynamic simulation are broken into individuals and expanded over the timeline as discrete events for the global scheduler. The global scheduler recognizes the expanded rounds as its initialization events and combines them with normal communication network events to form a global event list. When the simulation starts the global scheduler processes, either a power system dynamic simulation round or a communication network event according to their timestamp orders. When a power system dynamic simulation round is finished, the power simulator is suspended until next round is processed by the global scheduler. In this framework whenever an interaction between power system and communication network is needed, the request can be processed immediately by the global scheduler without unnecessary wait time. Both errors in Fig.3 are eliminated in this framework.

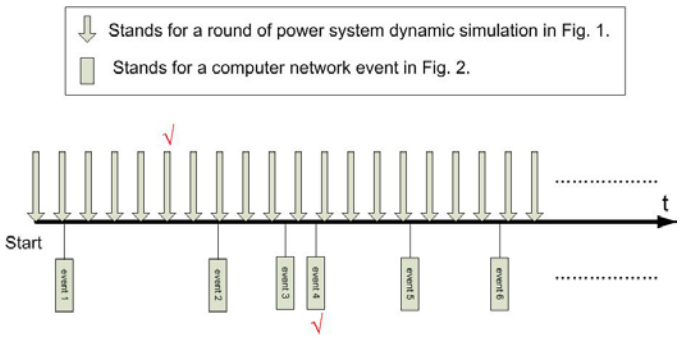


Figure. 4. Co-Simulation framework adopting a global scheduler without synchronization errors

We need to clarify that our co-simulation framework only removes accumulating errors introduced by synchronization between two running processes. It doesn't help to remove errors introduced by simulators themselves. If a big time step is selected for power system dynamic simulation it is still possible to have multiple network events happening during that interval and induce simulation errors. Also, if people use a very small synchronization interval in Fig.3 it converges to our co-simulation framework but with more simulation cost and limited time scale.

## III. Co-Simulation Implementation

We implement the preceding co-simulation framework by integrating two acknowledged simulators: Positive Sequence Load Flow (PSLF) software for power system dynamic simulation and Network Simulator 2 (NS2) for communication network simulation. The reason we choose these two simulators is that both of them are of proven simulation capabilities in their respective research domains, and have good supports for user defined extensions. It is better than writing simulation codes from the scratch as long as the selected software can be appropriately allocated in the co-simulation framework.

PSLF is a power system software package designed by GE which provides both steady-state and dynamic power system simulations. PSLF can simulate a system with up to 60000 buses and is equipped with a rich library of electromechanical dynamic models. The software suite is written in Java and provides application programming interfaces (APIs) for user for further development. User can interact with the simulator through a graphical user interface (GUI) or from a command line console. The software also comes with a script language EPCL which can configure user-customized simulation and add new dynamic models to the model library.

NS2 is an open source communication network simulation software developed by the academia, aiming at network performance evaluation. It is a well-known and widely used simulator in networking research domain because of its comprehensive support of network models. NS2 can simulates four layers in the Open Systems Interconnection (OSI) reference model for computer network protocols which include application layer, transport layer, network layer and data link layer. NS2 support most of the network protocols used in today's computer network and substantially covers wired, wireless and hybrid communication simulations. NS2 is mainly written in C++ and also provides a script language OTcl for simple and fast simulation configuration. A framework called "OTcl linkage" is realized in NS2 to cooperate the simulation written in C++ and OTcl. OTcl linkage establishes a class hierarchy mapping between C++ and OTcl so that whenever an object is instantiated in OTcl, a corresponding shadow object is created in C++. Both OTcl classes and C++ classes derive from the same father class TclObject. Through OTcl linkage, a user can write and compile new network protocols or models in C++ and configure the simulation by OTcl without repeating compilation for each simulation case. In this manner NS2 is nothing but an OTcl language translator.

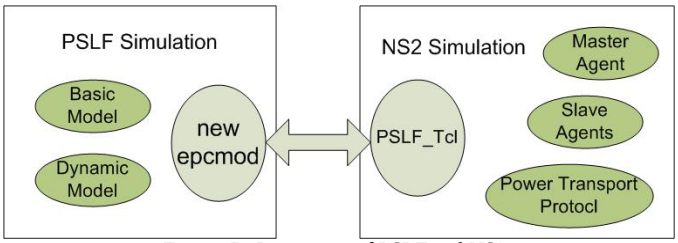


Figure. 5. Integration of PSLF and NS2

The co-simulation is realized by designing interface code on both sides as shown in Fig. 5. In NS2, its build-in network event scheduler is adopted as the global scheduler. A new class "PSLF_Tcl" derived from TclObject is designed in C++ as an interface to PSLF and initializes power system dynamic simulation rounds in NS2 for the global scheduler. This class sends command to PSLF interface model to collect power system data from PSLF after each dynamic round and stores them in NS2 for fast access by other network components. In PSLF, a new user-written dynamic model "epcmod" is attached to the simulated system. This model is designed as a stand-alone component in the power system and does affect the regular simulation. Its function is twofold: one is to explain commands from NS2 interface model and report back power system state data and another one is to suspend PSLF's simulation after each dynamic round and wait for NS2 commands to continue. The second function is crucial in our implementation. Without it the co-simulation cannot be scheduled by a global scheduler.

Other than the interface codes on both sides, a group of new network components are designed in NS2 to facilitate power system applications. A slave agent class is designed at the application layer in NS2. It stands for software or hardware agents interacting with power system and communication infrastructure in the real world. Typical instances of them could be the intelligent electronic devices (IEDs) or digital relays. The slave agents share a uniform design format so that they can be configured as different type of agents like bus agent, generator agent, relay agent etc., upon simulation requirements. A master agent is also implemented in application layer as a coordinator on top of all the slave agents. The master agent can stand for a central control module or an operation center. Furthermore several classic network transport protocols like User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) are modified so as to recognize and transmit power system data.

## IV. Agent-Based Remote Relay Backup Protection

A recently proposed agent-based supervisory remote backup relay protection scheme to prevent hidden failure based tripping is one example [9] whose validation requires a power system and networking co-simulation. Traditional relays monitor local system state and make autonomous decision. Even though current digital relays have been designed and equipped with comprehensive functionality and computation ability, in practice they are usually working on the lowest profile and most of the functions are not fully utilized. Hence there is great potential to upgrade the relay protection system without too much intrusion on their current structures.

In the agent-based supervisory remote backup relay protection scheme, instead of working individually, relays are connected with each other and co-operate their protection through extensive communication. Within the interested area, each distance relay has a software agent (slave) associated with it. This software agent works as an interface to other relays for information exchange. There is also a master agent in locale working as a global coordinator for other slave agents. In this form the system is able to provide better protection on the system and avoid hidden failure induced false tripping [10]. For example, whenever a relay picks up for a fault in its back up zone of protection or senses a change in the breaker status, the corresponding agent records it and sends the information to the master agent through the network. The master agent queries the other slaves in the vicinity of the reporting slave. Based on the responses from the other slaves, the master agent decides if there is a real fault or the reporting slave is picking up falsely. Accordingly, the master agent allows or blocks the tripping of that relay.

Fig.6 shows the typical communication between the master agent and slave agents. There are four steps in total before a decision is made for the backup relay: (1) Slave picks up a fault and submits a query to the Master; (2) The master collects information from other slaves; (3) Other slaves report back (4) Master sends decision back to the slave who submitted the query. It is necessary to study the total time delay consumed in these four steps to make sure the fault can be dealt with in time since protection is a time-critical application [11]. We show how to validate this scheme using our co-simulation framework.

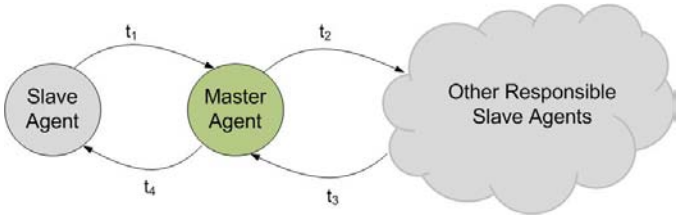


Figure. 6. Remote backup relay communication

## V. Co-Simulation Cases

We apply the agent-based remote relay protection scheme to the New England 39-bus system benchmark and simulate it on our co-simulation platform. The simulation case is studied not only to validate our co-simulation but also to evaluate the performance of the agent-based remote relay protection. We consider two conditions when the agent-based remote relay protection may take place: primary relay failure and remote backup relay false tripping. As shown in Fig.7 we construct the power system in PSLF and build a communication network in NS2 which has the same topology with the power system. The communication link is configured to have a bandwidth of 100Mbytes and 3ms communication delay on each link. Coaxial and microwave communication media usually has this representative settings.

### A. Case 1: Primary Relay Failure

We first assume a primary relay failure in the 39-bus system. A short circuit fault is created at 0.01s between bus 4 and bus 14. We also assume that the relay at bus 4 can normally see the fault and trip the breaker on its side but on the other side, the primary relay at bus 14 did not see the fault due to a failure or wrong relay settings. In this condition, the remote backup relay at bus 15 and 13 see the remote fault and request the master agent at bus 16 to make a decision. The master agent polls other relays and concludes that there is something wrong with the primary relay at bus 14. The command is then sent back to relays at bus 13 and bus 15. Here our decision making is simple: as long as there is another relay seeing the fault the remote backup relay should take action to protect the system since the chance that two relays have simultaneous malfunction is low.

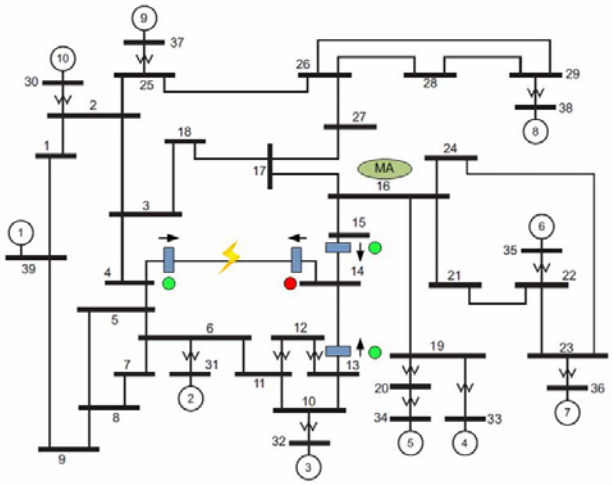


Figure. 7. Primary relay at bus 14 does not see the fault [12]

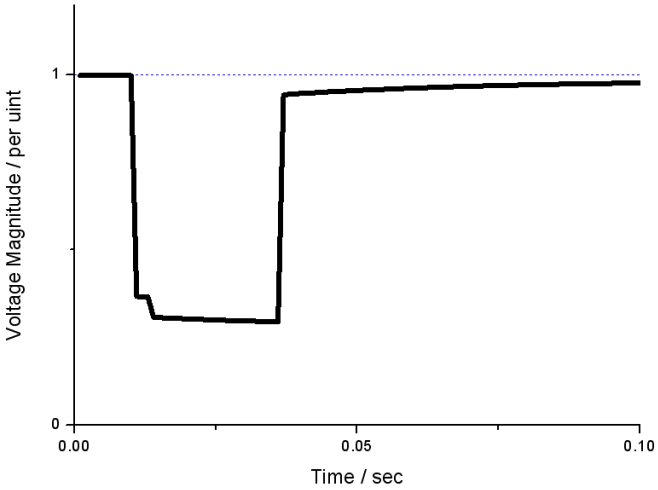


Figure. 8. Voltage magnitude at bus 13 in case 1

The simulated voltage magnitude at bus 13 is plotted in Fig. 8 from which we can see that after the fault happened the voltage drops and comes back after the fault is cleared at both sides. It roughly takes 25ms for the agent-based remote backup relay to clear the fault which is good enough to protect the system.

### *B. Case 2: Remote Backup Relay Failure*

We also consider a case of remote backup relay false tripping. A remote backup relay could see a false fault due to hidden failure. Without agent-based relay communication the backup relay could trip its circuit breaker and generate unwanted loss to the system. In our case, also in the 39-bus system we assume the remote backup relay at bus 13 sees a false fault owing to wrong relay settings. Then it sends a request to the master agent to collect information from others. The master agent checks other relays and cannot find anything wrong. Thereby the master sends a message to the relay at bus 13 telling that it is a false fault. Then the relay should stay still and report for maintenance. The voltage magnitude at bus 13 is again plotted in Fig.10 from which we can see that nothing has been changed in the system.

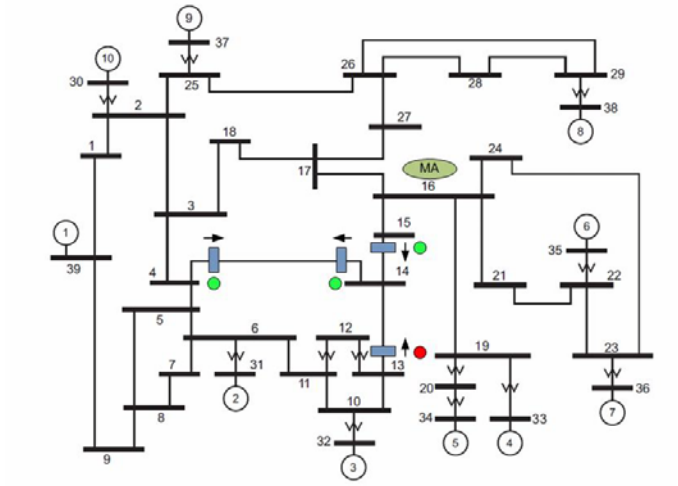


Figure. 9. Remote relay at bus 13 sees a false fault [12]

## VI. Conclusion

In this paper we propose a co-simulation framework which combines power system simulation and communication network simulation together. We analyze the continuous time model used in power system simulation and the discrete event-driven model used in communication network simulation. However, we leverage existing simulators for both the continuous state and discrete-event simulation by federating them. The previous federated simulators for the same purpose can be shown to have coarser grain synchronization induced inaccuracies. In this paper we show how to alleviate this problem by proposing an errorless implicit synchronization mechanism for the co-simulation framework. We further implement this co-simulation framework by integrating PSLF and NS2 software. An agent-based supervisory remote backup relay protection scheme is studied on this platform. Both the protection scheme and the co-simulation framework are validated this way.

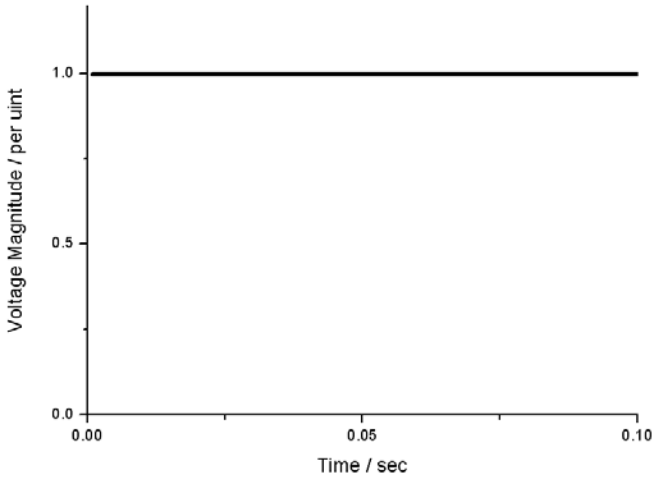


Figure. 10. Voltage magnitude at bus 13 in case 2

## VII. Acknowledgment

We would like to thank EPOCHS's author Dr. Kenneth Hopkinson for his advices on the co-simulation implementation.

## VIII. References

[1] U.S Department of Homeland Security, "National Power Grid Simulator Workshop Report," December 2008.

[2] C. M. Davis, *et al.*, "SCADA Cyber Security Testbed Development," in *Power Symposium, 2006. NAPS 2006. 38th North American*, 2006, pp. 483-488.

[3] N. Higgins, *et al.*, "Distributed Power System Automation With IEC 61850, IEC 61499, and Intelligent Control," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. PP, pp. 1-12, 2010.

[4] K. Hopkinson, *et al.*, "EPOCHS: a platform for agent-based electric power and communication simulation built from commercial off-the-shelf components," *Power Systems, IEEE Transactions on*, vol. 21, pp. 548-558, 2006.

[5] J. Nutaro, *et al.*, "Integrated Hybrid-Simulation of Electric Power and Communications Systems," in *Power Engineering Society General Meeting, 2007. IEEE*, 2007, pp. 1-8.

[6] F. Bouchhima, *et al.*, "A SystemC/Simulink Co-Simulation Framework for Continuous/Discrete-Events Simulation," in *Behavioral Modeling and Simulation Workshop, Proceedings of the 2006 IEEE International*, 2006, pp. 1-6.

[7] J. H. Taylor and Z. Jie, "Rigorous hybrid systems simulation with continuous-time discontinuities and discrete-time components," in *Control & Automation, 2007. MED '07. Mediterranean Conference on*, 2007, pp. 1-6.

[8] M. C. D'Abreu and G. A. Wainer, "Models for continuous and hybrid system simulation," in *Simulation Conference, 2003. Proceedings of the 2003 Winter*, 2003, pp. 641-649 Vol.1.

[9] S. Garlapati, *et al.*, "Agent Based Supervision of Zone 3 Relays to Prevent Hidden Failure Based Tripping," presented at the First IEEE International Conference on Smart Grid Communications, 2010.

[10] H. Wang and J. S. Thorp, "Optimal locations for protection system enhancement: a simulation of cascading outages," *Power Delivery, IEEE Transactions on*, vol. 16, pp. 528-533, 2001.

[11] J. W. Stahlhut, *et al.*, "Latency Viewed as a Stochastic Process and its Impact on Wide Area Power System Control Signals," *Power Systems, IEEE Transactions on*, vol. 23, pp. 84-91, 2008.

[12] I. Hiskens, "Significance of Load Modeling in Power System dynamics," presented at the x symposium of specialists in electric operational and expansion planning, 2006.