Simulating Smart Grid: Co-Simulation of Power and Communication Network

Abdul Razaq, Bernardi Pranggono, Huaglory Tianfield
School of Engineering and Built Environment,
Glasgow Caledonian University, Glasgow, UK
{ abdul.razaq, b.pranggono, h.tianfield } @gcu.ac.uk

Hong Yue
Dept. of Electronic and Electrical Engineering
University of Strathclyde, Glasgow, UK
hong.yue@strath.ac.uk

***Abstract*- This paper identifies challenges presented in unified co-simulation of power and communication network. There is no main stream freely available simulator which combines both power and network capabilities. First, review of existing simulators of smart grid is presented. Then, co-simulator scheme is proposed. Finally, a basic prototype of co-simulation based on open-source power and communication network is presented. Test cases show that our proposed co-simulator has faster execution time compared to standalone simulations.**

***Index Terms*—Co-Simulation, Smart Grid, Simulation**

I. INTRODUCTION

Smart Grid (SG) consists of two diverse technologies of power electronics and communications technologies that are integrated to form a cohesive and compelling system. It is a distributed system incorporating a multitude of various energy sources through energy management systems to stabilise demand and supply of electricity. In this paper, we identify challenges presented in SG that the ideal simulation system should address. Our objective is to identify the state of art and provide a design to develop Smart Grid Simulator.

In general, there are two kinds of simulation technologies, namely, physical simulation and computer based simulation. Physical simulation test-beds for power networks are not convenient to access by a vast number of researchers and are difficult to scale which include hardware cost, laboratory space and safety equipment. Computer based simulation systems are cost-effective and flexible to replicate complex systems without being involved in actual architecture while only acquiring knowledge of software application. Computer based co-simulation is an indispensable design component of SG which allows investigating every aspect of the hazardous system without adverse consequences. A malfunction in such a delicate system can cause severe consequences including interruption of electricity, equipment damage, data breach, complete blackouts or even life threatening consequences.

Section 2 reviews existing work in the literature on various techniques to co-simulate SG with power and communication systems. It also discusses open-source solutions that can be utilized to perform SG simulation. Section 3 presents a co-simulation scheme for smart grid. Section 4 provides discussions.

II. LITERATURE REVIEW

There are a number of Power and Network simulators with open and commercial licenses. However, recently there has been a trend for a proven and open-source co-simulator which combines power and communication technologies.

A. Physical Simulation and Computer Based Simulation

Physical simulation based on power system hardware-in-the-loop is a common technique. However, this involves actual physical components. As indicated by Tan et al. [1] that physical simulation test-beds are not convenient to access by vast research community and are difficult to scale. Besides, experimental investigation of complex and large systems is usually not economically feasible because of physical infrastructure. It is important, to note, that the multilevel cost of these physical devices is a major hurdle which includes cost of components deployment expenses, laboratory space with adequate equipment and safety concerns because of the hazardous nature of AC. In addition, the majority of the community involved in SG comes from different knowledge domains, namely Software Applications, Cyber Security, Networking, Automation and Communication. Physical simulation test-beds make it difficult for these researchers to study the Smart Grid with respect to their domain focus.

Computer based simulation systems are cost-effective and flexible to replicate complex systems without being involved in actual architecture while only acquiring knowledge of the software application. Computer based simulation results are also easier to analyze than typical experimental results because it is relatively easy to log and trace important information at critical points in order to diagnose the system's behavior. Existing simulation systems can be used to some extent to simulate SG sub-systems; however, most of these simulators are not purpose built.

B. Co-Simulation of Smart Grid

Co-simulation is a way to couple two or more different subsystems to form a unified modeling and simulation system. SG co-simulation consists of power and communication simulators. Several attempts have been made to combine power network and communication network simulators based on commercial and open-source solutions. Some have adopted open-source network solution with commercial power simulator, whereas, others have utilized open-source power simulator with their own communication implementation or opens-source solution.

The work by Hopkins et al. [2] can be tracked back as introducing one of the early simulators to combine realistic network communications with electric power components. Their work was based on ns-2 and commercial off-the-shelf components. Electric power and communication synchronizing simulator (EPOCHS) was introduced for distributed simulation environments. Mets et al. [3] have theoretically evaluated different solutions to co-simulate SG. They have included commercial and open-source simulators for network and power domains. The simulators are classified according to targeted use cases, model level of detail, and architecture. A similar research conducted by Li et al. with focus on SG's communication provided an overview of challenges and categorized these in terms of impact on the simulation results [4].

Dugan et al. [5] demonstrated a hypothetical example using an Open Distribution System Simulator (OpenDSS) to model the power distribution system and a ns-2 Network Simulator to model communications network. Open-source ns-2 is a discrete event simulator that provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless networks. Dugan et al. [5] hypothetical system is based on work by Godfrey et al. [6] which consists of power and network simulators built on an OpenDSS and ns-2. Lin et al. [7] also demonstrated the identical approach while utilizing the ns-2 and GE's Positive Sequence Load Flow (PSLF). Their global event-driven co-simulation framework 'GECO' integrates the simulations of power system and communication network to achieve synchronization accuracy.

Chassin et al. [8] utilized GridLAB-D (GLD) and MatPower as its backend for power simulation. Yan et al. [9] investigated the communication interface of SG with ns-2 and OMNET++, these network simulators are open-source and widely respected in the research community. Both solutions have been used in cyber security domains and are the best candidate for simulation systems for SG cyber security. Hence, the existing network security solutions can be utilized where they are adequate and new dedicated solutions can be developed to meet the SG specific challenges where traditional enterprise network cyber security solutions do not work or apply.

The work conducted by Anderson et al. [10] named as GridSpice is an interesting attempt in SG simulation. They have utilized GLD and MatPower as its backend for power simulation. GridSpice provides an intriguing user friendly graphical user interface (GUI) and an option of scripting for advance users [11]. Tan et al. [1] have practiced similar approaches and have developed ScorePlus simulator for cyber physical test-bed that addresses the intelligent control, communication, and interactions in SG. They have created Advanced Metering Infrastructure (AMI) network with IEEE PES 37 test feeders. The simulator termed as SCORE is written in C language which is portable to virtual nodes. GLD is being used as a power simulator with home-developed communication modules; this solution lacks a proven network simulator and security is not explicit focus. Bhar et al. [12] presented a co-simulation with widely used power and network simulators. They have integrated OpenDSS and OMNET++ with COM interface termed as DSS Solver developed in VC++. Authors' claim that time synchronization is resolved while providing a framework for continuous and event-based requirements. A similar work was conducted by Sun et al. [13] while using OpenDSS as the power simulator and OPNET for network simulation. The interaction between these two simulators is achieved with introduction of intercommunication module which is developed in MATLAB. Caire et al. [14] developed co-simulator with PSAT from MATLAB for power simulation and ns-2 for Information and Communications Technology (ICT). Both simulators communicate over UNIX style named pipes and time synchronization is managed by a component called coordinator developed in Java. Coordinator uses a fixed time-slot for time-stamping the system. This solution lacks open-source power simulator.

A very intuitive work by Zhou et al. [15] resulted in InterPSS as a fully open source platform for Power System Simulation (PSS). However, InterPSS has limited features for AC load flow analysis and feasible for limited distributed systems. They claimed to have simulation algorithms; user interfaces (UI), input and out modules based on the same plugin design. InterPSS is a promising development for PSS domain, however, it falls short for integrated network simulation.

Chinnow et al. [16] extended the InterPSS with Network Security Simulator (Nessi). Their main area of focus is Smart Metering or AMIs while proving the security paradigm for such infrastructure. Mets et al. [17] adopted a similar approach but with a different toolset. They proposed an integrated framework that simulates both the communication network and power networks. OMNET++ was the tool of choice for the communication part and commercial MATLAB product was used to model the distributed grid for PSS. Gomes et al. [18] simulated partial functionality of SG based on agent-based design for demand and response. They claim to have developed their own solution implemented in C# and Java programming languages. This solution is suitable to understand electricity consumption and distribution characteristics of SG but it is not capable to simulate generation and transmission.

Most recently, Ciraci et al. [19] presented an interesting co-simulator termed as FNCS based on GLD and ns-3. FNCS is implemented in C++ with interfaces for C, Java and Fortran. It also implements the publisher-subscriber design of HLA to provide customization and integration. Authors claim to overcome the time synchronization problem between two simulators through a two-phase synchronization scheme that synchronizes before and after processing a time step. The proposed algorithms presented here [20] and implemented on FNCS framework provide improvements over the existing algorithm. This simulator is at a very initial stage without security as its focus, however, has a great potential to be a streamline SG simulator. Table 1 presents classification of SG simulators.

Table 1 - SG CO-SIMULATORS

Name	Power Simulator	Network Simulator	Cyber Security	Simulation As Service	Customization (HLA)	Open-source
Hybrid Simulator	**OpenDSS**	**ns-2**	**N/A**	**N/A**	**N/A**	**Yes**
EPOCHS	**PSLF**	**ns-2**	**N/A**	**N/A**	**Yes**	**Partial**
GECO	**PSLF**	**ns-2**	**Yes**	**N/A**	**N/A**	**Partial**
Gridspice	**GridLAB-D**	**N/A**	**N/A**	**Yes**	**Yes**	**Partial**
ScorePlus	**GridLAB-D**	**N/A**	**N/A**	**Yes**	**Yes**	**Yes**
Co-Simulator Framework	**OpenDSS**	**OMNET++**	**N/A**	**N/A**	**N/A**	**Yes**
Co-Simulation Platform	**OpenDSS**	**OPENT**	**Yes**	**N/A**	**N/A**	**Yes**
InterPSS	**Limited Features**	**N/A**	**N/A**	**Yes**	**Yes**	**Yes**
Nessi2	**Limited**	**Limited**	**Yes**	**N/A**	**N/A**	**Yes**
Integrated Simulation	**MATLAB**	**OMNET++**	**Yes**	**N/A**	**N/A**	**Partial**
FNCS	**GridLAB-D**	**ns-3**	**N/A**	**Partial**	**Yes**	**Yes**

As presented in Table 1, most of the solutions do not allow customization, simulation as a service and lack cyber security. FNCS is the best solution in this list with all features except cyber security and partial ability to offer simulation as a service. However, FNCS is also the only solution to use ns-3 as network simulator which is newer version and not widely used as ns-2.

III. PROPOSED CO-SIMULATION SCHEME

In this section, we present a co-simulation scheme and demonstrate our proposed techniques to co-simulate power and ICT network within a single application with efficient execution time. Power and network architecture is presented with specifications of physical components and link layouts with functional configurations.

A. *Simulation Architecture*

Simulation solutions such as ns-2 and OMNET++ are the best communication network simulators with open-source and widely respected in the research community. On the other hand, GridLab-D and OpenDSS are equally proven and verified open-source implementations for power grid simulation. Our proposed co-simulation uses ns-2 for ICT and GLD as power simulator, both solutions are open-source and freely available. ns-2 is a discrete event simulator which provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless networks. Tool Command Language (TCL) script is used to create models with required objects for a specific simulation scenario. Output of simulation can be user defined or default formats of x-y data plotter (Xgraph) and Network Animator (NAM). ns-2 is a model based simulation tool. Each model in ns-2 consists of at least one model which is further configured depending upon required simulation. A model defines the type of network topology, traffic load, applications types and specification of each node in the network. Multiple models can be defined in a single TCL file. Power simulator GLD is also model based and it was developed by the U.S. Department of Energy (DOE) under funding for Office of Electricity in collaboration with industry and academia. GLD offers distribution automation design, peak load management via various peak-shaving methods and distributed generation/storage. Models in GLD are provided in the form of GLM file format and output is configured to be in user-defined format, Open Database Connectivity (ODBC) or matlab formats. GLM files are used to synthesize populations of objects and encode object behaviour in GLD.

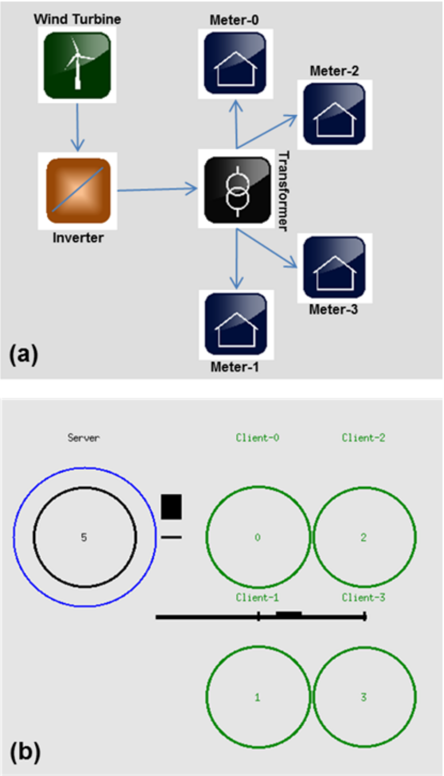

Fig. 1. High level architecture (a) power network, (b) communication network.

Fig. 1 presents the high level architecture of co-simulation for both networks. ICT network is illustrated in 1(b) with corresponding power network illustrated in 1(a). ICT system consists of local area network (LAN) topology configured with a server connected to four nodes labelled as Client. A parallel power network consists of single wind turbine generator and four consumption nodes labelled as Meter. Wind turbine generates constant power and is connected to an inverter which eventually connects to a transformer. Power grid simulates power generation from the wind turbine and voltage levels on meter nodes. IP model simulates the TCP communication network for ftp application. There is no one-to-one mapping in power and ICT nodes.

B. *Co-Simulation Prototyping*

Our research aim was to investigate complete open-source solution to co-simulate SG within single application interface. This consolidates the workload and improves simulation time. The experiment was performed on a machine running fedora version 20 with 64-bit architecture with pre-emptive mode, supported with 12 GB RAM and Intel Pentium 2020M CPU.

Integration of GLD and ns-2 was accomplished at very early stages while coupling both the independent and whole applications in a single parent process. These both utilized solutions are open source; hence, derived work is available for further analysis [21]. ns-2 provides an entry point to start the co-simulation. A new agent called 'AgentGL' was developed which is a bare-bone agent derived from ns-2's agent class. AgentGL is a C++ application which issues ‘exec’ system call in newly forked process and wait for its return. The forked process executes GLD with GLM file as input which is provided by TCL script at the beginning of simulation. AgentGL has the capability to be called from TCL script and capable to invoke all available ns-2’s functionalities. AgentGL offers published methods that are visible from TCL scope. In order to run GLD simulator, the user should create a GLD's GLM script file, and invoke the method 'callexecute-gl' within TCL script.

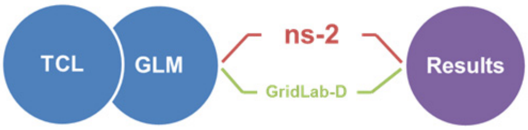

Fig. 2. Workflow of proposed co-simulator.

Fig. 2 illustrates how both simulators are combined together in parallel to execute models and yield results. Input is based on a TCL script file which also includes path of a GLM script file, AgentGL accepts this TCL file and starts the ns-2 for network simulation before starting GLD for power simulation. AgentGL will fork a process with ‘exec’ system call and wait on ns-2 along GLD.

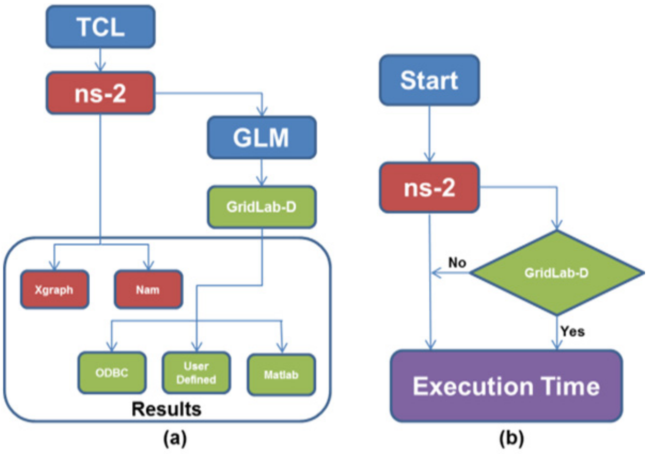

Fig. 3. Co-simulator (a) execution architecture. (b) Execution time calculation of co-simulator.

Fig. 3 illustrates the execution of co-simulation. TCL script of ns-2 initiates simulation which contains network model details and file location to GLM script which contains power model. Simulator ns-2 will parse the input script file and start network simulation process; next, it will start the GLD process with GLM file as an input model which was provided by TCL script. Both simulators will provide output results into specified format as instructed by individual models.

There are two different techniques to execute the GLD and ns-2 scripts from AgentGL. 1) User can use AgentGL's 'call-execute-ns' method similar to GLD's 'call-execute-gl'. 2) Or user can define ns-2 model within the TCL script as usual and invoke 'call-execute-gl' for GLD. Time of execution is calculated via shell script which records the starting time before executing the application, and finally the time is recorded when application makes an exit. Difference between exit and start time provides the total execution time of tested application. It’s important to note that there is almost constant overhead time which is because of shell script, however, this is presented in all test cases. Execution time can be verified via cross check from raw nano-second with calculated time in seconds. See Fig. 3-(b).

C. *Test Cases*

This embedded solution provides a single interface and unified technique to simulate ns-2 and GLD. This work in progress highlights the novelty and usefulness of co-simulation. However, the presented approach does not significantly achieve complete unification but serves as a starting point for further development. Similar attempts have been made as reviewed in the literature review section; however, the presented solution is unique with respect to simplicity, openness and unification. It is efficient in terms of execution time compared to the usage of individual simulators.

Three test cases were investigated with sample sizes of 1000 for each test-case. Tests include 1) Execute ns-2 as standalone, 2) Run GLD as individual application and 3) Execute ns-2 and GLD from developed AgentGL. Fig. 4 provides the average execution time for all three test cases.

Fig. 4. Average execution time for 1000 test cases of ns-2, GLD, Agent and combined average time of ns-2 with GLD

These individual test cases were executed as independent applications. Results noticeably points to the advantage of unified solution with least execution time marked as ‘Agent’ compared to the average combined time of ns-2 and GLD. This combined time is obtained individually, for each application as discussed in above paragraph, and does not include the time it would take to switch from one application to another. In case of human effort, it will be constant but very noticeable when executing two separate applications manually. If proposed method is used to execute both applications from a single interface then it will not only reduce the human effort but also consolidate the simulation with minor overhead of third shell process. Our approach overcomes these challenges while modifying the ns-2 to fork a new process for GLD with its separate process space; thus, allowing users to observe both simulations simultaneously.

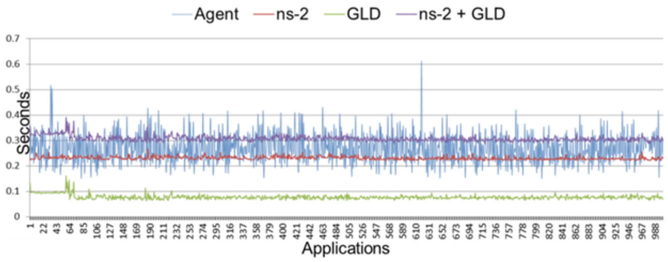

Fig. 5. Actual execution time for 1000 test cases of ns-2, GLD, Agent and combined average time of ns-2 with GLD.

The performance efficiency of the developed solution is evident in Fig. 5 with a sample of 1000 tests for each test-case. Execution time with our 'Agent' solution is visible with slightly less execution time compared to aggregated time of ns-2 and GLD. This also reduces the constant overhead by half. These results could be improved if coupling of applications is completely replaced with a single application. Single application in terms of single process or execution point would definitely require merging the two code bases. Although this will reduce the process level fork, wait and IPC overhead but radically increase the system complexity. Complexity usually introduces task dependencies that are normally achieved via threading or so called signalling mechanism; thus further introducing wait in process runtime. Coupling two applications independently and yet providing unified experience proves to be practical in terms of code management and performance. Our proposed design follows similar approach while applications being independent but connected on higher level to form coupling.

IV. DISCUSSIONS

SG simulation is challenging. Unfortunately, there is no main stream freely available Simulator which combines both power and network capabilities based on proven solutions. We have presented an overview of existing SG simulators and critical analysis based on deployment, usage and licensing limitations. FNCS is promising direction towards SG co-simulation with dedicated messaging system and time synchronization. However, it is in its very early stage of development, and ns-3 is being used as communication network simulator which is not widely used as ns-2. It also lacks logical mapping between power and network domains.

As described above none of the existing open source SG Simulators has addressed co-simulation properly, and many of them are used are with very limited scope and difficult to expand for specific scenarios. Moreover, existing simulation solutions are quite complicated in terms of deployment and usage. This limitation restricts the simulation usage only to experts in power or network fields with in-depth knowledge of these individual simulators. Besides, these solutions require resourceful provision because of resource complexity involved in SG simulation. Researchers from relevant domains of SG are unable to resolve associated problems because of unavailability of SG Simulator which can offer simulation as simulation only without indulging into complexity and bearing recurring cost.

The performance efficiency of our proposed co-simulation solution is evident. It confirms the fact that co-simulation is definitely a better solution compared to standalone solutions. However, co-simulation also faces challenges in integrating two different domains while synchronizing the time requirement between power and network systems. A messaging mechanism is also required for inter-process or task communication. Finally, logical mapping from one domain to another is important to carry out true co-simulation of SG.

V. FUTURE WORK

Combining different domains with a unified solution presents real challenges. Hence, a solution must be generic to accommodate subsystem requirements and overlapped areas where these components interact to form a cohesive SG system. Co-simulator should be generic to subsume different models and produce results for a single SG system.

The major challenge for co-simulator is to synchronize the time in two different domains. Both simulation systems require a different timing and how they handle and respond to time is also different. Power system dynamics is commonly simulated in a continuous time series. However, communication network simulation is usually performed using a discrete event-driven method.

Another challenge in co-simulation is mapping the entities from one domain to another. This includes creating relationships between components based on specific usage case. These relationship mappings could be one-to-one (e.g, meter to LAN client), one-to-many (e.g, transformer to Building Area Network) and many-to-many (e.g. micro-gird to LANs). For example, in our current presented example, every single meter node should be a corresponding LAN client in ICT network. It would allow real-time linkage between power and ICT components. For example, meter load can be monitored by the LAN client.

In order to achieve linkage between these two domains it further requires a messaging mechanism in place. This would allow the passage of messages back and forth to or from power and ICT components. Co-simulator should provide a unified way to send and receive messages from one simulator to another simulator.

REFERENCES

[1] S. Tan, W.-Z. Song, D. Huang, Q. Dong, and L. Tong, “Distributed Software Emulator for Cyber-Physical Analysis in Smart Grid,” *Emerg. Top. Comput. IEEE Trans.*, vol. PP, no. 99, p. 1, 2014.

[2] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, L. Fellow, K. Birman, and D. Coury, “EPOCHS : A Platform for Agent-Based Electric Power and Communication Simulation Built From Commercial Off-the-Shelf Components,” vol. 21, no. 2, pp. 548–558, 2006.

[3] K. Mets, J. A. Ojea, and C. Develder, “Combining Power and Communication Network Simulation for Cost-Effective Smart Grid Analysis,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 3, pp. 1771–1796, 2014.

[4] W. Li, M. Ferdowsi, M. Stevic, S. Member, A. Monti, S. Member, and F. Ponci, “Cosimulation for Smart Grid Communications,” vol. 10, no. 4, pp. 2374–2384, 2014.

[5] R. Dugan, E. Us, S. Mullen, T. Godfrey, and C. Rodine, “21 st International Conference on Electricity Distribution HYBRID SIMULATION OF POWER DISTRIBUTION AND COMMUNICATIONS NETWORKS 21 st International Conference on Electricity Distribution,” no. June, pp. 6–9, 2011.

[6] T. Godfrey, S. Mullen, R. C. Dugan, C. Rodine, D. W. Griffith, and N. Golmie, “Modeling Smart Grid Applications with Co-Simulation,” *Smart Grid Commun. (SmartGridComm), 2010 First IEEE Int. Conf.*, 2010.

[7] H. Lin, S. Member, S. S. Veda, S. S. Shukla, S. Member, L. Mili, J. Thorp, and L. Fellow, “GECO : Global Event-Driven Co-Simulation Framework for Interconnected Power System and Communication Network,” vol. 3, no. 3, pp. 1444–1456, 2012.

[8] D. P. Chassin, K. Schneider, and C. Gerkensmeyer, “GridLAB-D: An open-source power systems modeling and simulation environment,” in *Transmission and Distribution Conference and Exposition, 2008. T #x00026;D. IEEE/PES*, 2008, pp. 1–5.

[9] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, “A Survey on Cyber Security for Smart Grid Communications,” *IEEE Commun. Surv. Tutorials*, vol. 14, no. 4, pp. 998–1010, 2012.

[10] K. Anderson, J. Du, A. Narayan, and A. El Gamal, “GridSpice: A distributed simulation platform for the smart grid,” *2013 Work. Model. Simul. Cyber-Physical Energy Syst.*, pp. 1–5, May 2013.

[11] K. Anderson, J. Du, A. Narayan, and A. El Gamal, “GridSpice: A Distributed Simulation Platform for the Smart Grid,” *IEEE Trans. Ind. Informatics*, vol. 10, no. 4, pp. 2354–2363, Nov. 2014.

[12] D. Bhor, K. Angappan, and K. M. Sivalingam, “A co-simulation framework for Smart Grid wide-area monitoring networks,” in *Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference on*, 2014, pp. 1–8.

[13] X. Sun, Y. Chen, J. Liu, and S. Huang, “A co-simulation platform for smart grid considering interaction between information and power systems,” *Isgt 2014*, pp. 1–6, Feb. 2014.

[14] R. Caire, J. Sanchez, and N. Hadjsaid, “Vulnerability analysis of coupled heterogeneous critical infrastructures: A Co-simulation approach with a testbed validation,” in *Innovative Smart Grid Technologies Europe (ISGT EUROPE), 2013 4th IEEE/PES*, 2013, pp. 1–5.

[15] M. Zhou and S. Zhou, “Internet, Open-source and Power System Simulation,” in *Power Engineering Society General Meeting, 2007. IEEE*, 2007, pp. 1–5.

[16] J. Chinnow, K. Bsufka, A.-D. Schmidt, R. Bye, A. Camtepe, and S. Albayrak, “A simulation framework for smart meter security evaluation,” *2011 IEEE Int. Conf. Smart Meas. Futur. Grids Proc.*, pp. 1–9, Nov. 2011.

[17] K. Mets, T. Verschueren, C. Develder, T. L. Vandoorn, and L. Vandevelde, “Integrated simulation of power and communication networks for smart grid applications,” in *Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2011 IEEE 16th International Workshop on*, 2011, pp. 61–65.

[18] L. Gomes, P. Faria, H. Morais, Z. Vale, and C. Ramos, “Distributed, Agent-Based Intelligent System for Demand Response Program Simulation in Smart Grids,” *Intell. Syst. IEEE*, vol. 29, no. 1, pp. 56–65, Jan. 2014.

[19] S. Ciraci, J. Daily, J. Fuller, A. Fisher, L. Marinovici, and K. Agarwal, “FNCS: a framework for power system and communication networks co-simulation,” in *Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative*, 2014, p. 36.

[20] S. Ciraci, J. Daily, K. Agarwal, J. Fuller, L. Marinovici, and A. Fisher, “Synchronization Algorithms for Co-simulation of Power Grid and Communication Networks,” in *Modelling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), 2014 IEEE 22nd International Symposium on*, 2014, pp. 355–364.

[21] A. Razaq, “Co-Simulation of Power and Network,” *GridLab-D And ns-2*, 2014. [Online]. Available: https://github.com/arazaq/ns2. [Accessed: 16-May-2015].