Towards generic SCADA simulators: A survey of existing multi-purpose co-simulation platforms, best practices and use-cases

Kostas Mathioudakis, Nick Frangiadakis, Andreas Merentitis, Vangelis Gazis
AGT Group (R&D) GmbH, Hilpertstrasse 35, 64295 Darmstadt, Germany
Email: {kmathioudakis, nfrangiadakis, amerentitis, vgazis}@agtinternational.com

***Abstract*—SCADA (Supervisory Control and Data Acquisition) systems monitor industrial and critical infrastructures, so assessment of their performance and security is important. They are complex systems deployed over wide areas, hence their modeling and simulation is far from trivial. In fact, often it necessitates the co-simulation of the processes supervised, as well as the network communications between SCADA components. Selecting the appropriate modeling and simulation tools is critical, because the combination of domain specific simulators with network simulators while considering the crucial aspect of time synchronization, enables the deployment of dependable simulation environments. This paper describes the architecture of SCADA networks, analyzes the characteristics of the most common used network simulators and surveys existing simulator implementations in multiple SCADA application domains. As SCADA systems have evolved towards standardization, there are great opportunities for simulation tools that will allow us to enhance and fine-tune their performance. The key objective of this paper is to review the state of the art of performance simulation of SCADA simulation systems.**

Keywords*—SCADA system architecture, SCADA networks, SCADA security, co-simulation environments, network simulators

I. Introduction

SCADA systems [1], [2] have become the paramount technology for monitor and control of large-scale industrial and critical infrastructures during the last decades. They are consistently used in oil and gas refineries and pipelines, water treatment and distribution, electrical power generation and transmission, as well as in railways. The term SCADA is commonly used to describe computerized control systems whose field devices are geographically spread, sometimes beyond the country borders. These field devices are connected to their control center(s) via Wide Area Networks (WANs) using diverse protocols, over wired or wireless channels.

SCADA systems typically cover wide area and are comprised of multiple, often redundant entities as well, as communication modes between them. They are deployed over, and expected to remain operational for decades providing high availability. As a result, they are quite complex while needing to adhere to strict availability, robustness, extendability and often security constraints. SCADA simulators are used to help with many of those tasks, including planning, resource optimization, personnel training, security checks, demand prediction, and robustness to disaster scenarios. Legacy deployments often involve non-standard architectures and custom-designed software. Modern deployments adhere to Industry standards, however, due to high requirements, even they may be highly customized.

Standardization of SCADA systems and especially of SCADA networks has also boosted the reusability of components for SCADA simulators. Towards this direction the High Level Architecture (HLA) [3] facilitates the development of simulator environments, providing the generic framework to combine multiple simulation platforms. However, when designing any simulator it is crucial to take into account where and how it is going to be used. This paper provides a survey of existing SCADA simulator tools, but also aims to serve as a guide of what are the best practices one should keep in mind when using those tools.

The rest of the paper is organized as follows: In Section II we describe the architecture of the SCADA systems giving simultaneously a quick overview of the communication topology and the security issues. In Section III we highlight the requirements and the architecture that a co-simulation platform should be in line. In Section IV we describe and analyze the communication network co-simulation tools as well as we survey the most relevant research implementations. Finally, we conclude in Section V by briefly describing the most critical challenges that have to be tackled in order to develop a reliable and robust SCADA simulation environment.

II. SCADA Systems

Because SCADA are control systems where availability is key requirement, they often involve more or less tight closed control loops that allow the system to operate for large time periods even in the absence of external interaction.

Distributed Control Systems (DCS) on the contrary, are smaller, closed loop systems that cover a limited area and geographically localized processes not exceeding the bounds of an installation. DCSs are also used in industrial applications to monitor and control distributed equipment without human intervention for its normal operation. For large operations, DCS is operating as part of SCADA and is interconnected with it, thus allowing operator interaction. The field equipment is connected to the DCS control center via a local area (LAN) control network, offering higher reliability as well as relatively higher physical and cyber security compared to a flat SCADA architecture.

In the early days of SCADA systems, they run on dedicated networks supporting only proprietary protocols with limited connectivity. Security was based on physical location separation as well as the offered by the obscurity of vendor-specific software and hardware. Among other imposed limitations, this cause scalability as well as interoperability problems. However, most modern gateway devices support multiple standard protocols over IP and are often wireless. Modern SCADA systems usually offer robust communications utilizing multiple communication channels with varying reliability, throughput, and latency characteristics. Detailed and fast control of multiple locations allows for faster and more precise reactions that fine tune the system operation. However, at the same time the complexity of the system increases.

1) Architecture: A typical SCADA architecture is depicted in Figure 1. Field devices such as Remote Terminal units (RTUs) or Programmable Logic Controllers (PLCs) are connected with sensing equipment (meter readings) and switch-boxes or valve actuators spread in the process field. The RTUs in remote locations accumulate data and send them via a communication link to the SCADA master station. The SCADA server compiles the data, transfers alarms and events to the Human Machine Interface (HMI) and stores the collected data in a large database called historian. The historian logs and archives time-based process data allowing performance monitoring, trend analysis and auditing. The HMI is used to provide an interface to the operators. It displays data to the operators and allows them to monitor the state of the process and to interact with the field devices providing input forms, when an action is required. Multiple HMIs may exist with diverse access rights depending on the requirements of different users (e.g. operator or system engineer).

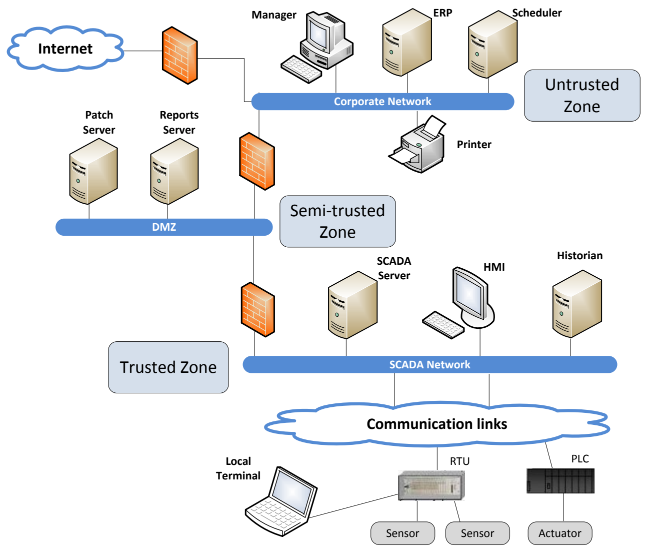

Fig. 1: SCADA Architecture

The top layer as illustrated in Figure 1 is the corporate infrastructure layer. This includes Enterprise Resource Planning (ERP) systems for operational management, production scheduling, business planning and logistics. Usually, the business management layer contains various servers, hosts, network devices providing internet services such as web, FTP, and e-mails. Moreover, this layer usually supports remote user access points in order to allow distant users to examine the state of the SCADA system. Although this is not recommended for security reasons, some authorized and privileged remote users could interact with the SCADA control devices. Nevertheless, the corporate infrastructure layer is considered untrusted due to its exposure to the internet. Therefore, remote access should be restricted only in emergency situations.

2) Local Communication: The communication between the field devices and the control center is provided by the control network. Historically, SCADA networks have been dedicated networks using proprietary protocols. That led to complex types and structures with no support for interoperability. Dozens of those proprietary protocols exist. Nevertheless, over the years the industrial control companies started adopting common open standard protocols and after the proliferation of IP networks the trend is to encapsulate SCADA packets into TCP/IP frames. The most popular industrial protocols are Ethernet/IP, Modbus-TCP, DNP3, Profibus etc. Furthermore, to address interoperability through the creation and maintenance of non-proprietary open standards specifications, the OPC Foundation [4] developed an Application Programming Interface (API); the OPC protocol. OPC facilitates the data transfer between the industrial control systems, the HMI, the historian and the Enterprise Resource Planning (ERP) systems. More recently, the platform independent OPC UA protocol [5] was introduced in order to disengage from the Microsoft COM based OPC classic.

3) Security: SCADA progressively moves towards general purpose information technologies such as Ethernet and TCP/IP for both critical and non-critical communications. However, while the use of common protocols is considered beneficial and cost effective, it exposes the vital operational processes to threats coming from the outside world. Several types of attacks ranging from unauthorized user access (hacking) and eavesdropping to data interception and denial of service attacks expose SCADA systems in high risk. The best defense against these threats is to completely isolate the network from the outer world. However, this cannot be entirely done; therefore additional means of enhancing security [6] should be considered. In the direction of mitigating these cyber threats, one commonly suggested practice is to isolate the SCADA and process control networks from the enterprise network and the internet through the deployment of firewalls, creating different zones of trust. Architectures that allow the communication between the untrusted enterprise networks to the trusted SCADA network only via Demilitarized Zones (DMZ) provide the most effective security solution as illustrated in Figure 1. It has been long advocated to apply the principle of least privilege which provides a user or process only the minimum set of rights absolutely required to perform a task whilst deny everything else. Additionally, when the architecture requires the deployment of multiple firewalls, it is highly recommended to use firewalls from different vendors in order to enhance the security.

III. EMULATION AND SIMULATION SUPPORT

The main difference between a common IT infrastructure and a SCADA system is the operational continuity that a SCADA system must support. This means that even the most secure, trusted and reliable update cannot be immediately applied without rigorous testing in part of the real system. Any potential malfunction that occurs during the installation of new software may interrupt the critical operation process causing loss of availability and consequently significant damage to the operation. Even more, as SCADA coupled with command and control systems are used for monitoring of vital operations they cannot be directly used for other activities such as patch management and software testing, cyber security testing, network performance evaluation or personnel training.

Even though the procurement of a full redundant system for addressing such non-operational activities is most probably the ideal solution, it may not be the feasible. A SCADA system is comprised of several units in diverse domains and that makes the deployment of redundant equipment impossible or cost ineffective. Taken this into account the deployment of a reliable simulation environment that adheres to all the SCADA requirements is the most promising way for addressing the aforementioned issues.

In addition, the interaction of communication networks and the domain specific systems such as electric grid components would be cumbersome to study without the aid of accurate models and scalable simulation environments. The analysis of complex processes which include integrated, discrete or continuous methods require appropriate simulation models and techniques. In this context, power grid modernization efforts call for powerful modeling and simulation tools for hybrid systems. Furthermore, in power grid applications (e.g substation automation) or more general in wide area monitoring scenarios the use of modern communication technologies is considered mandatory; hence, the adoption of cutting edge network simulators is the only feasible way to experiment and evaluate these networks. For these reasons, it is essential to develop a simulation framework that will allow for modeling and simulating the different components in various circumstances.

a) General Simulation Requirements: A SCADA simulator should be composed of simple but reusable components and should support extensibility and easy interconnection with other simulating and/or real modules. At the same time, it should not introduce more complexity than required for the task at hand. As we will discuss in paragraph IV-A, choosing the right level of and tools for simulation, affects the correctness and the efficiency of our model. The best choice depends on the specific system the SCADA is controlling, the expected scenarios, the rate of changes in the system and their relative speed to that of network communications, as well as the accuracy we want to obtain.

b) Architecture: As we describe in the following paragraphs a general architecture of a SCADA simulation environment is constituted from four diverse layers as illustrated in Figure 2. In the bottom we meet the domain specific models or real world field devices if we target supporting "hardware in the loop". In the second layer there is a thoroughly designed coordination module which is responsible for the interconnection as well as the critical time synchronization between the models and the network simulator. The latter lies on the third layer while on top there is an interface that allows the user to interact with the developed platform. Usually the user interface is incorporated in the network simulation software.

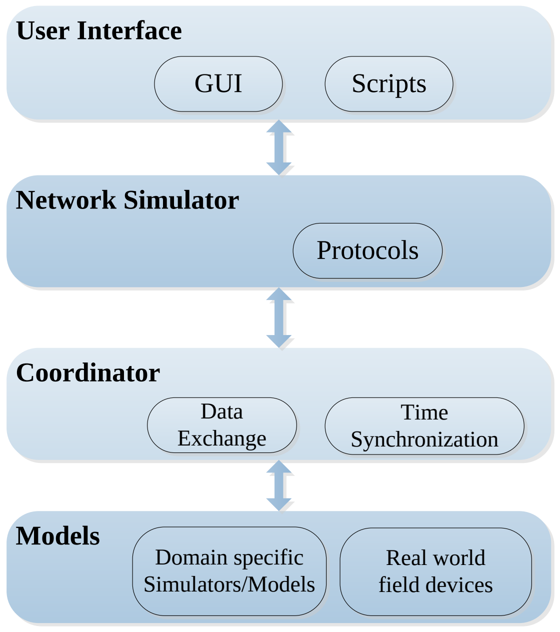

Fig. 2: Design structure of SCADA simulation environment

IV. SIMULATORS

Developing a SCADA simulator can be time-consuming and costly. Due to the complexity of the problem and the various aspects of an industrial control system, a single simulator is not enough to simulate the whole SCADA process. Neither a stand-alone domain specific simulation nor a stand-alone network simulation is adequate to model a fully operational and interconnected system. Consequently, there is a significant research effort to combine different domain specific and network simulators into co-simulation platforms. IEEE's High-Level-Architecture (HLA) step towards this direction as we describe below.

A. *Simulation Model Considerations*

When using a simulator, our goal is to describe in adequate detail the state of a system, that is a collection of entities [7]. We should keep in mind that this is not the only way we can go about and in fact other methods involve experimenting with the actual system, using a physical model of the system on an analytical solution. For example, to simulate network traffic we can choose to use a simple connectivity matrix, unless realistic metrics for delay and latency are needed, in which case a packet level network simulator would be more appropriate.

Simulating a SCADA system involves modeling at given levels of detail the state of the SCADA entities, the communications between them and the state of the supervised / controlled operation (henceforth called 'environment'). Creating a monolithic simulator for all these would not only be costly, but also would produce inefficient results, since selecting the right level of detail for each of those depends on our requirements. In fact, some systems may have a part of them not simulated, but emulated or partially implemented using a physical model. As an example, a simulator used for planning future needs or approximating operational costs may not need to detail day-to-day operations on high level. However, a simulation used to demo a particular capability, or to train personnel might actually have partially implemented the aspects of the SCADA system that interact with personnel.

Furthermore, when designing a simulator, we should consider how the environment states interact with each other. Consider as an example the cases of a SCADA for an electric grid, versus that of one for monitoring a set of dykes. In the first case, any change in the electric load will affect the system in nano, micro, or miliseconds depending on the phenomenon. In the second, an anomaly may be noticed some days before the dyke actually breaks. This means that a simulator of the 'environment' may in the first case need to have a nanosecond detail, while in the second minutes or hours. Now consider a typical communication network where events have milisecond duration. Combining it to the 'environment simulator' may not make much sense. It may be better to run the two simulations separate and provide a connectivity matrix for communications, or a set of possible connectivity states and their Probability Distribution Functions (PDFs).

B. High Level Architecture (HLA)

Introduced by the US Department of Defense, HLA [3] was developed to ensure the interoperability and reusability of models and simulation components. It is a standard method to allow for combining distinct simulations (federates) into a combined co-simulations platform (federations). Its aim is to provide a structure, which will enable the reuse of some capabilities already available in other simulations, targeting cost and time reduction.

A computer simulation or a manned simulator can act as a federate, which is represented as object. Additionally, the Runtime Infrastructure (RTI) acts as the distributed operating system of the federation. It is a collection of services that support the interactions between different federates. Its major service is considered the time management which goal is to ensure the time synchronization among all federates. Several commercial and open source RTIs are available. Finally, the runtime interface specification describes how the interaction between federates and the RTI is conducted.

Each individual simulator should conform to HLA's rules and interface specifications in order to be combined with other federates. Moreover, each federate or federation document should be openly available and conform to the standard open model template. Unfortunately, it may be complex to modify already existing simulations to comply with HLA's specification. Although there is a possibility to add a wrapper to these simulators, it is cumbersome due to the number of modifications and additions that should be done in order to conform to the HLA.

C. Network simulators

Network simulators in general try to mimic and model real world networks. The cost of a complete implementation of an industrial communication network just for testing and analyzing is prohibitive compared to the development of a simulation framework. Although network simulators are not perfect, they can give a meaningful insight into the tested network, and how changes will affect its operation. Below, it follows a short description of the most commonly used network simulators.

c) NS2: The Network Simulator 2 is an open source event-driven simulator designed for research in computer communication networks. It can simulate existing network protocols such as TCP, UDP, routing and multicast protocols over both wired and wireless networks. It allows the creation of various communication scenarios using specific protocols and simulating the behavior of these protocols under diverse conditions. NS2 utilizes C++ to define the simulation objects and Object-oriented Tool Command Language (OTcl) to schedule the discrete events and to set up the simulation. NS2 is particularly relevant to industrial simulations since there is a trend to encapsulate proprietary protocols packets into TCP running over Ethernet networks.

d) NS3: Network Simulator 3 has borrowed concepts and implementations from several open source simulators and it is considered that will progressively replace NS2. However, NS3 is not an updated version of NS2 and it is not backward compatible with NS2. It is designed to improve scalability and modularity and is written is C++ but offers also a Python scripting interface. Moreover, it supports virtualization, software and testbed integration, and it focuses its attention to realism supporting key interfaces such as sockets and network devices, multiple interfaces per node and use of IP addresses.

e) OPNET: OPNET simulator is a tool to simulate the behavior and performance of any type of network. It is based on discrete system event mechanism which simulates the system behavior by modeling the events of the scenarios that the user has set up. Its main feature is that it provides various real-life network configuration capabilities that make the simulation environment close to reality. It also provides graphical editors (GUI interface) and a comprehensive library of network protocols and models. It employs object-oriented programming techniques to create the mapping from the graphical design to the implementation of the real systems. Finally, it allows the development of user's own networks, protocols and packet formats by providing the necessary programming tools and documentation.

f) OMNeT++: OMNeT++ is also a discrete event simulator programmed in C++. It is a component based architecture and consists of modules that communicate with each other using message passing. The components (programmed in C++) can also be combined in a hierarchy of levels allowing the creation of complex simulation components. It additionally provides a GUI interface and due to its modular architecture, the simulation kernel can be embedded into all kinds of different users' applications. Moreover, by providing plug-ins extensions, it allows the modification of the default behavior of the simulation engine (e.g. different message scheduler).

D. SCADA simulator frameworks

The integration of different simulation frameworks for heterogeneous systems is common in research. However, simulating SCADA systems and networks using open source tools is relatively new. The reason for that is due to the lack of proper modeling tools, the communications characteristics have to be simplified and simulations have to be based on significant assumptions that may affect the final outcome. Table I summarizes the related works in this direction.

The authors in [8] exploit the OPNET capabilities for simulating the communication network and they use the Virtual Test Bed (VTB)[1] for dynamic simulation of the power system. The VTB is a software tool that used for power electronics and power systems. For the data exchange between the two simulators, a co-simulation coordinator was developed which also is responsible for the time synchronization. The coordinator provides a user interface and through it the user can set the global time step. The overall goal of this effort is to study and analyze the communication network of the power system and to check the stability of the power system as a function of the network performance. Their simulation results show significant issues in delay and packet dropping when they examined the sampling period and the communication data rates.

In [9] the authors propose a SCADA simulation tool (SCADASim) that supports the integration of external devices (e.g. smart meters or RTUs) and applications. Their objective is to examine the effect of attacks in real devices and applications by using a simulated environment. Attacks that are supported include denial of service, man in the middle, eavesdropping, and spoofing. Moreover, they aim to build an extensible, flexible and modular SCADA simulation framework with simultaneously integration of external components and devices. They use the OMNeT++ to simulate the network and they exploit the socket based integration of OMNeT++ to allow the integration of the external devices. However, in order to tackle the issue that only socket-based protocols are supported, they deploy gates. These gates act as communication ports that implement protocols for communicating with external components. The synchronization between the OMNeT++ and the external devices is handled by the SSScheduler module, which is responsible for synchronizing the corresponding clocks. Finally, they deploy malicious attacks (denial of service and spoofing) scenarios to evaluate the framework, and they demonstrate how the attacks are affecting the process of use legitimate requests.

In [10] the authors propose a reference architecture. It consists of several layers and components that represent the enterprise network, the OPC server and client, the SCADA protocols tester, the RTUs, the field sensors and actuators as well as the industrial infrastructure. Their prototype implementation targets the security analysis and assessment of SCADA systems. It is extensible and adaptable and it is mainly based on NS2. Moreover, in order to allow the integration with real networks they exploit the capabilities of emulation feature of NS2. The latter has the ability to inject traffic from the simulator into a live network and to simulate a desired network between real applications in real-time. For the simulation framework they use real PLC/RTUs and sensors/actuators, as well as industrial and open source systems for OPC client/server implementation. Finally, for the evaluation of the simulation framework they implemented attack scenarios that compromise the security of SCADA system and they developed methods to analyze and assess the impact of these attacks on the system. Nevertheless, there is no much information available regarding the outcome of the experiments.

In [11] the authors integrate multiple research and commercial-of-the-self (COTS) systems to build an agent based simulation framework for the electric power grid. Based on HLA framework [3] they created a combined simulation system which exploits the capabilities of several of the self simulators; a combination of the PSCAD (Power System Computer Aided Design) and EMTDC (Electromagnetic Transients including Direct Current) [18] where the first offers a graphical interface and the latter is an electric power simulator, the PSLF (Positive Sequence Load Flow) for elecromechanical transient simulations, used also in elecromechanical stability scenarios, and the NS2 for the network communication simulation. All these simulators conform to the HLA's rules, interface specification, and documentations standards. Additionaly, the interface between the individual simulators is performed by a central component, the RTI. The RTI is responsible for packet routing and the time-stepped synchronization between the federates. This synchronization method is the most commonly used when multiple simulators are cooperating. A preset simulation time has to be reached by all the components before a data exchange between them is allowed. Apart from the diverse simulators the authors developed the AgentHQ module which acts as a proxy when agents need to interact with the federates. Through it, it is possible to set and get the power system values and to exchange data. Finally, they demonstrate a number of experimental results showing the pros and cons of an agent based special protection schemes, while they do not deal with security issues.

In [12] the authors inspired by [11] and [17], developed a co-simulation framework targeting smart grid applications. Their contribution was the alleviation of the synchronization overhead problem that could cause data mismatch between the simulation components, by introducing "asynchronous" approaches. Their primary focus is to remove the accumulating errors introduced by the synchronization mechanism. For this reason they run the simulation globally in a discrete event-driven manner rather than using continuous time simulation methods to simulate a discrete event system. A global scheduler is responsible for the synchronization while the different simulators share the same timeline. For the implementation they leverage the capabilities of the PSLF simulator for power system dynamic simulation and the NS2 for the communication network simulation. Moreover, an agent-based relay protection scheme is examined within this framework. Each relay has its own master and slave agent which act as interface for data exchange. Finally, for the validation they just examine relay failures without dealing with security issues.

The authors in [13] designed the simulation platform DAVIC (Distributed Automation Via Implicit Channels) which is based on OMNeT++. The objective of this implementation lies on the creation of fast and reliable simulation platform to be used for the evaluation of different energy management algorithms such as peak demand. Interestingly, they did not use of-the-self domain specific simulators, but they used synthetic load profiles which are the reference consumption profiles, as well as their own calculations in order to model the demand. That saved them from synchronization difficulties but limited the usability of the platform.

[1] http://vtb.engr.sc.edu/vtbwebsite/#/Overview

TABLE I: Simulations frameworks

	Network Simulator	Domain Specific Simulator	Synchronization	Domain
VPNET [8]	OPNET	VTB (Virtual Test-Bed)	Timed stepped (co-simulation coordinator)	Communication Networks of Power Systems
SCADASim [9]	OMNeT++	Built-in modules	SSScheduler module	Malicious attacks in SCADA systems
Wang Chunlei et al. [10]	NS2 (NSE)	CitectSCADA 6.1 (as OPC server) & Real PLC/RTUs	Not mentioned	Security analysis of SCADA systems
EPOCHS [11]	NS2	PSLF, PSCAD/EMTDC	Timed stepped	Electric Power Grid
Lin et al. [12]	NS2	PSLF	Timed stepped (global scheduler)	Smart Grid Applications
DAVIC [13]	OMNeT++	Built-in models	Time-stepped	Energy management algorithms
Chabukswar et al. [14]	OMNeT++	Simulink	Timed stepped (NetworkSim Sceduler)	Attacks on SCADA systems
Neema et al. [15]	OMNeT++	X4 Simulink models	Timed stepped (NetworkSim Sceduler)	Computational experiments in Command & Control
Naturo et al. [16] [17]	NS2	ADEVS	DEVS	Electric Grid

The objective in [14] is to demonstrate the use of C2WindTunnel [19], [20] platform with the aim to simulate DDOS-like attacks on a plant and its control system as well as to analyze the effects on different routers. The C2WindTunnel platform is based on HLA and it was designed to facilitate the development of large-scale simulations. It uses the Generic Modeling Environment [21] and employs model-based design techniques and graphical interface to allow integration of diverse simulation engines. The authors use the NetworkSim, which is based on OMNeT++ to simulate the communication protocols and the Simulink to model the domain specific processes. Moreover, they developed a Simulink function to synchronize the model with the Run-Time Infrastructure allowing the Simulink to progress only when the RTI allows it. They use timed-stepped synchronization, while keeping the time-size low in order to minimize event timing errors introduced by exchanging events between Simulink and HLA.

Another work that is based on the C2WindTunnel is presented in [15]. This work outlines the challenges encountered in deploying simulation platforms that are used to mimic the command and control environments. As most of the previous works the authors used OMNeT++ and in order to preserve the synchronization between the OMNeT++ and the RTI they developed, which called the NetworkSim scheduler. For their experimentation they implemented Unmanned Aerial Vehicles (UAVs) models using the Simulink X4 with the objective to test the mission performance when a network attack is occurred. Especially, they focused on DDOS attacks describing the consequences on the quality of received data and the communication between the command and control center and the UAV.

In [16] and [17] a hybrid simulation tool was developed for modeling the communications and the control of the electric grid. The primary objective of the authors is to preserve the hybrid simulation definition by leveraging both discrete and continuous systems. The communication processes are modeled using NS2 and the discrete and continuous processes are implemented using ADEVS (A Discrete EVent Simulator). ADEVS is a C++ library for constructing discrete event simulators based on parallel DEVS and Dynamic DEVS formalism. The ADEVS software is encapsulated in an NS2 TclObject and it is invoked by the NS2 when required. In the experimentation phase they show how the communication network affects the order of load shedding as well as how the bandwidth and the latency affect the controller behavior.

E. Simulator software license

Most of the software packages that were used by the researchers are free or at least an academic license is available. Nevertheless, some tools were built targeting industrial or commercial applications, thus the procurement of a license is considered necessary. The following Table II summarizes the license requirements of the software packages that were mentioned in paragraph IV-D.

TABLE II: Software license

	Type	License
NS2	Network Simulator	Open Source
NS3	Network Simulator	Open Source
OMNeT++	Network Simulator	Free (available open source simulation models)
OPNET	Network Simulator	License required
Simulink	Model-Based Design	License required
VTB	Power System Dynamic Simulator	Free
PSCAD/EMTDC	Power System Transient Simulator	Free version available
PSLF	Power System Analysis Software	License required

V. Conclusion

Several challenges are faced when developing a heterogeneous simulation platform for SCADA. Although the HLA framework provides the basic APIs to mitigate the complexity of developing simulations, there are still issues that have to be tackled during the development phase. Three levels of integration are often required in order to incorporate a simulation framework to an overall simulation environment; the API level, the level of interactions, and the level of model semantics. The first offers some basic services such as management functionality and message passing. The second and most important manages the time synchronization and coordination among the complex command and control simulation platforms, while the third is rather optional and depends on the objective of the simulation environment. Although many of the self simulation frameworks provide some of the required services and are compatible with the HLA reference architecture, they lack an overarching integration and coordination approach among multiple platforms.

Minimizing the effort and time required for simulation development can be accomplished through integration of multiple appropriate domain-specific tools, which will lead to more adaptive SCADA simulation environments. Concluding, the incorporation of already deployed and validated large-scale domain specific models is a key prerequisite for the development of powerful hybrid systems.

References

[1] Axel Daneels and Wayne Salter, "What is scada," in *International Conference on Accelerator and Large Experimental Physics Control Systems*, 1999, pp. 339–343.

[2] Zhendong Ma, Paul Smith, and Florian Skopik, "Towards a layered architectural view for security analysis in scada systems," *arXiv preprint arXiv:1211.3908*, 2012.

[3] Judith S Dahmann, Richard M Fujimoto, and Richard M Weatherly, "The department of defense high level architecture," in *Proceedings of the 29th conference on Winter simulation*. IEEE Computer Society, 1997, pp. 142–149.

[4] "http://www.opcfoundation.org," last access: July 2013.

[5] Sebastian Lehnhoff, Sebastian Rohjans, Mathias Uslar, and Wolfgang Mahnke, "Opc unified architecture: A service-oriented architecture for smart grids," in *Software Engineering for the Smart Grid (SE4SG), 2012 International Workshop on*. IEEE, 2012, pp. 1–7.

[6] Keith Stouffer, Joe Falco, and Karen Scarfone, "Guide to industrial control systems (ics) security," *NIST Special Publication*, vol. 800, pp. 82, 2008.

[7] Averill M Law and W Kelton, "David: Simulation modeling and analysis," *Mac Graw Hill, Boston, Burr Ridge, ua*, 2000.

[8] W Li, A Monti, M Luo, and Roger A Dougal, "Vpnet: A co-simulation framework for analyzing communication channel effects on power systems," in *Electric Ship Technologies Symposium (ESTS), 2011 IEEE*. IEEE, 2011, pp. 143–149.

[9] Carlos Queiroz, Abdun Mahmood, and Zahir Tari, "Scadasim - a framework for building scada simulations," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 589–597, 2011.

[10] Wang Chunlei, Fang Lan, and Dai Yiqi, "A simulation environment for scada security analysis and assessment," in *Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on*. IEEE, 2010, vol. 1, pp. 342–347.

[11] Kenneth Hopkinson, Xiaoru Wang, Renan Giovanini, James Thorp, Kenneth Birman, and Denis Coury, "Epochs: a platform for agent-based electric power and communication simulation built from commercial off-the-shelf components," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 548–558, 2006.

[12] Hua Lin, Santhoshkumar Sambamoorthy, Sandeep Shukla, James Thorp, and Lamine Mili, "Power system and communication network co-simulation for smart grid applications," in *Innovative Smart Grid Technologies (ISGT), 2011 IEEE PES*. IEEE, 2011, pp. 1–6.

[13] Peter Palensky, Friederich Kupzog, Adeel Abbas Zaidi, and Kai Zhou, "A simulation platform for distributed energy optimization algorithms," in *Industrial Informatics, 2008. INDIN 2008. 6th IEEE International Conference on*. IEEE, 2008, pp. 342–347.

[14] Rohan Chabukswar, Bruno Sinópoli, Gabor Karsai, Annarita Giani, Himanshu Neema, and Andrew Davis, "Simulation of network attacks on scada systems," in *Proceedings of the First Secure Control Systems Workshop, Cyberphysical Systems Week, Stockholm, Sweden*, 2010.

[15] Himanshu Neema, H Nine, G Hemingway, J Sztipanovits, and G Karsai, "Rapid synthesis of multi-model simulations for computational experiments in c2," 2009.

[16] James Nutaro, Phani Teja Kuruganti, Mallikarjun Shankar, Laurie Miller, and Sara Mullen, "Integrated modeling of the electric grid, communications, and control," *International Journal of Energy Sector Management*, vol. 2, no. 3, pp. 420–438, 2008.

[17] James Nutaro, Phani Teja Kuruganti, Laurie Miller, Sara Mullen, and Mallikarjun Shankar, "Integrated hybrid-simulation of electric power and communications systems," in *Power Engineering Society General Meeting, 2007. IEEE*. IEEE, 2007, pp. 1–8.

[18] Olimpo Anaya-Lara and E Acha, "Modeling and analysis of custom power systems by pscad/emtdc," *Power Delivery, IEEE Transactions on*, vol. 17, no. 1, pp. 266–272, 2002.

[19] Karen E Roth and Shelby K Barrett, "Command & control wind tunnel integration and overview," in *Proceedings of the 2009 SISO European Simulation Interoperability Workshop*. Society for Modeling & Simulation International, 2009, pp. 45–51.

[20] Graham Hemingway, Himanshu Neema, Harmon Nine, Janos Sztipanovits, and Gabor Karsai, "Rapid synthesis of high-level architecture-based heterogeneous simulation: a model-based integration approach," *Simulation*, vol. 88, no. 2, pp. 217–232, 2012.

[21] Akos Ledeczi, Miklos Maroti, Arpad Bakay, Gabor Karsai, Jason Garrett, Charles Thomason, Greg Nordstrom, Jonathan Sprinkle, and Peter Volgyesi, "The generic modeling environment," in *Workshop on Intelligent Signal Processing, Budapest, Hungary*, 2001, vol. 17.