Towards a Design Studio for Collaborative Modeling and Co-Simulations of Mixed Electrical Energy Systems

Yogesh D. Barve, Himanshu Neema , Stephen A. Rees and Janos Sztipanovits
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN 37212, USA

***Abstract*—Despite the known benefits of simulations in the study of mixed energy systems in the context of smart grid, the lack of collaboration facilities between multiple domain experts prevents a holistic analysis of smart grid operations. Current solutions do not provide a unified tool-chain that supports a secure and collaborative platform for not only the modeling and simulation of mixed electrical energy systems, but also the elastic execution of co-simulation experiments. To address above limitations, this paper proposes a design studio that provides an online collaborative platform for modeling and simulation of smart grids with mixed energy resources.**

***Index Terms*—Smart Grid, Modeling, Co-simulations, High-level Architecture, Cloud Computing, Docker, Distributed Simulations, Model Driven Engineering, Experimental Testbed, Computer simulation, Systems simulation, Collaborative work , Power system simulation, Cyber-Physical Systems**

I. Introduction

With a rapid growth in mixed energy generation technologies such as wind and solar energy, the generation and distribution of energy is moving from a centralized grid to a more distributed paradigm. Traditionally, in the energy grid, the flow of energy distribution was from the main power grid operator to the consumers. However, due to increasing affordability of renewable energy harvesting equipment such as solar panels, there is an influx of energy produced by traditional utility consumers, who are feeding the excess energy back to the power grid. The presence of these new actors, also known as prosumers, will only accelerate as the cost of generating energy from these distributed energy resources (DER), such as solar, wind, and biomass, further decreases with advancement in efficient solar panels, energy storage technologies, and related power systems techniques.

With the rise in these mixed sources of energy, the future smart grid systems need to manage distribution of the power flow around the system. The distribution system operators (DSO), thus will play a crucial role and will need to intelligently manage the power demand and supply balance. To provide the best service, a DSO needs to aggregate real-time information about the local power demand and as such needs resilient communication infrastructure to read from all the smart energy meters and to have control on the network flows. DSOs can also set billing rates depending on the power demands. Market analysis thereby becomes more significant to the DSO who will want to set dynamic pricing for the power, in order to accommodate fluctuation in energy demands on a daily and seasonal basis. Since DSOs also have the authority to set prices to buy and sell power from the local prosumers, the role of market regulator also becomes crucial, so as to not have any DSO monopoly over the power market. Market regulators are thus responsible for maintaining transparency and competitiveness in the local energy market.

Thus, in the mixed energy smart grid system there are many stakeholders, who are directly responsible in successful operation of the smart grid system. Figure 1 illustrates some of these stakeholders in the system. As can be seen, for a successful smart grid system, domain experts representing various stakeholders need to holistically analyze and study the system.

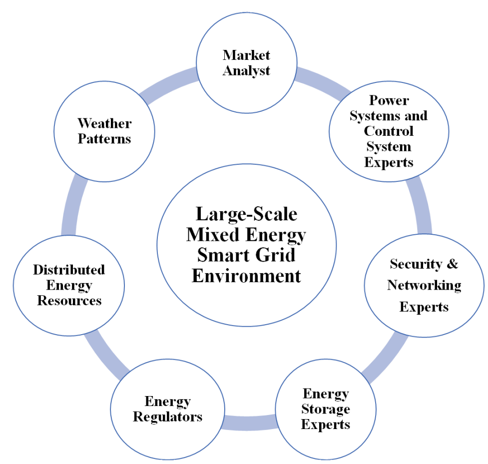

Fig. 1: Stakeholders in a Mixed Energy Smart Grid System

An approach to studying such large-scale complex systems is by means of simulation. However, smart grid systems are composed of multi-domain subsystems comprising security, DER, cyber-communications, control systems and electric grid. Thus, one needs a simulation and analysis tooling infrastructure that spans these multiple domains for designing and simulating such large-scale systems. Despite advances in the simulation tools, a single simulation tool is not able to capture and simulate these various physical and other system aspects of these multi-domain systems. Thus *co-simulation* or coupled simulation have gained popularity in bringing together simulation tools of different domains to simulate scenarios as in the case of mixed energy smart grid system.

As there are multiple domain experts who would need to design and analyze such complex system, there is a need for efficient tools and facilities that enable *real-time collaboration* between these participating actors. In addition, as these domain experts work on business and sensitive processes who may want to keep their work *secure* and *private*, these collaboration tools must have a secure private storage for all collaborators.

Large-scale smart grid simulation execution is a highly compute- and data-intensive activity. Thus, a large pool of computing resources are required to support *high performance computing* of these simulations.

To address the above requirements – *co-simulation, real-time collaboration, security, privacy, and high performance computing*, we present, in this paper, a design studio for collaborative modeling and co-simulation of mixed energy electrical systems.

The rest of the paper is organized as follows. Section II provides a background on the co-simulation platform, the collaboration platform, the collaborative modeling environment, and experiment execution in the cloud. Section III describes the design and implementation of the design studio. Section IV presents concluding remarks and future work.

II. Background

In this section, we will briefly cover the four fundamental technologies that enable our design studio. These include: (i) the co-simulation platform, (ii) the collaboration platform, (iii) the collaborative modeling web-bench, and (iv) simulation executions in the cloud.

A. Co-Simulation Platform

Owing to the lack of simulation tools' ability to cover multi-domain simulation characteristics, co-simulations offer an excellent alternative to analyze and simulate such multi-domain simulations. Co-simulation approach integrates different domain-specific simulators to provide a cohesive platform for carrying out studies for scenarios such as the mixed energy systems simulation for smart grids. *High-level architecture* (HLA) [1] provides a standardized method to integrate different simulators and execute them as a co-simulation. HLA provides information, synchronization, and coordination services among participating simulators. In HLA terminology, each of the participating simulator is called a federate. Different federates communicate with each-other by exchanging data according to their described publish and subscribe relationships. These different federates are time-synchronized – a crucial service for a distributed simulation. C2WT [2], developed at Vanderbilt University, is a heterogeneous simulation integration framework. It provides a model-based integration technology for rapid synthesis of distributed co-simulations such as those required for multi-domain simulations of cyber-physical systems (CPS). C2WT relies on the HLA standard and utilizes its open-source implementation (or Run-Time Infrastructure (RTI)) called Portico [3]. In this work, we are building on top of the C2WT framework to support cloud-scale simulations of mixed energy electrical systems in the context of smart grids.

B. Collaboration Platform

The Cyber-Physical Systems Virtual Organization (CPS-VO) [4] is a collaborative, web-based portal developed to promote interaction between academia, government, and industry across multiple disciplines in the burgeoning field of CPS. CPS-VO provides facilities for enabling repeatable, verifiable, shareable experiments and results to the users. CPS-VO supports multi-user collaboration communities with additional experimentation cloud, tools configuration, and a testing framework, which enables rapid development of collaborative design solutions.

To support this, the CPS-VO has a three key elements: tool libraries, integrated tools, and design studios. Tool libraries are searchable repositories of available software categorized by multiple taxonomies. Integrated tools are software solutions that are embedded within the CPS-VO and ready to be utilized without having to setup a server, download, install, or configure anything. Embedded tools run in the CPS-VO cloud in order to retain the elasticity to accommodate changing demand while maintaining the security and stability of the CPS-VO. These capabilities of the CPS-VO are actively being utilized for CPS education. As more tools are being added to the CPS-VO, these elements are being increasingly utilized by a diverse set of communities.

For our purposes, we will be employing each of these features – listing the completed tool in the library, providing a design studio interface for designing multi-domain co-simulation experiments, as well as integrating a tool for executing these experiments. Results from the experiments will be retrieved from the cloud by the CPS-VO and securely made available to the user, with an ability to download or delete as needed.

C. Collaborative Modeling Web-Bench

Modeling is an important phase in the design of an experiment. For co-simulation experiments, modeling enables domain experts to design parameterized simulation models for studying how they perform when they interact with other simulation models participating in the co-simulation.

For collaborative modeling of co-simulation experiments, a modeling environment should support the following: (1) To enable multiple experts to collaborate and participate in building the simulation models, the platform should support collaborative modeling. (2) The modeling environment should be able to support real-time editing and synchronization of simulation models across different participating domain experts. (3) Modeling environment should provide intuitive visual interface so that it lowers the entry of barrier to utilizing the new environment. This enable users to focus more on developing models rather spending time in learning a new toolsuite. (4) Provide tools for checking correctness of models and flagging constraint violations during designing of large-scale multi-model simulation experiments.

Taking these considerations into account, we selected the WebGME [5] modeling environment developed at Vanderbilt University. WebGME provides a web-based design and modeling environment. WebGME enable users to leverage model driven engineering techniques (MDE) [6] to develop large-scale software systems [7]. It provides facilities such as ability to create a visual domain specific modeling languages (DSML) using metamodeling. It allows creating model interpreters that are linked with the metamodels. Model interpreters enable automated software synthesis in the form of code artifacts and configurations, which are used by the domain experts to write simulation models and business logic to be embedded in the simulation models. WebGME supports checking model correctness and ensuring its conformity to set of constraints – a crucial while designing large-scale simulations. Visual notifications are shown for design time violations.

D. Cloud-Hosted Experimentation Platform

Large-scale smart grid simulation models are highly computation- and data-intensive. Thus, simulation models execution can benefit from the large resource pool provided by the cloud computing model. Cloud computing provides an on-demand access to these compute resources for running simulation execution experiments. Cloud computing has been leveraged for running large-scale simulation execution [8], [9], [10] in understanding and studying architectures for building smart-city scale distributed systems.

Despite the advancement in the cloud computing systems, the research community is faced with numerous challenges in moving their simulation models to the cloud computing environment. Executing in the cloud computing environment needs understanding of the cloud-oriented configuration and deployment tools [11]. Insufficient expertise in these tools can lead to performance degradation of the executing processes [12]. Another challenge in using cloud computing is the difficulty in migrating simulation execution models from the desktop or laptop based execution platform to the distributed and scalable cloud platform environment. Another important consideration when running execution in the cloud environments is to avoid getting tied to a single cloud provider or what is called as 'vendor lock-in'.

To address these challenges related to the deployment of simulation executions in the cloud environment, we leverage open-source technologies such as Openstack cloud hosting [13] and linux container based Docker technology [14] to host simulations. We will cover more details in the next section.

III. Design and Implementation of Design Studio

In this section we will cover the design and implementation of the collaborative platform for modeling and simulation of the mixed energy smart grid simulations.

A. Design And Architecture

Many building blocks are required to support collaborative modeling, simulation, and execution of the co-simulation experiments. Figure 2 showcases the main components of the design studio platform. These components are described in detail below.

1) CPS-VO: As discussed in II-B, the CPS-VO provides an entry portal to the users of the design studio. It provides users and user groups management functionality. This provides features such as user authentication and permission management to enable secure and private collaboration among users within a group or community. User access to the community portal is managed through a highly customized Drupal PHP based web portal. Access to the WebGME based modeling environment and the experimentation portal is channeled via the CPS-VO portal. When transitioning from simulation modeling to running an experiment, it facilitates access to the run-time cloud infrastructure. Currently, the CPS-VO provides a naive experiment scheduling policy by restricting the number of concurrent experiments to the total available execution computing nodes. Experiment results are automatically retrieved by the CPS-VO after the experiment has finished, and are stored and made available according to the privacy settings the user has selected, combined with group settings.

2) Simulator Federate Templates: This component provides access to various types of federates which are simulator specific. These includes C++, Java, Gridlab-D, and OMNET++ simulation engines. Users can build scenarios utilizing these simulation engines to construct large-scale simulations. Using these templates, one can construct an integration model of the simulated scenario. This integration model represents different federate type entities participating in the simulation. The integration model also covers any interactions, shared objects that may be exchanged between participating federates.

3) Courses Of Action Models: To conduct scenario-based experimentation and conducting what-if analysis [15], we support a modeling construct called as Courses-of-Action (COA). COAs are utilized to create various what-if analysis models and to execute the corresponding alternative scenarios. COAs act as an orchestrator of the time-coordinated execution of the running simulations. These COAs are scenario models that are created using several atomic elements such as: *ACTION* – that injects an interaction into the running simulation, *OUTCOME* – that waits for an interaction of the specified type to be generated in the running simulation, *FORK* – that start multiple branches in a scenario to start in parallel, and *DUR* – that, when encountered in a COA execution, makes a running simulation wait for the specified duration. A COA model is created as a workflow like Directed Acyclic Graph (DAG) by connecting the above-mentioned atomic elements with directed edges. Detailed description of many other supported COA elements can be found in [9].

4) Experiment Scenarios Models: Once the simulation integration model is designed with the constituent simulation federates, the data model for the data exchange among federates, and the objects that capture various actors in the co-simulation, the next step involves creating the experimental scenario models. Experimental model enables creating scenarios that comprise either some or all the federates from the integration model created in the previous step. As such, for a given co-

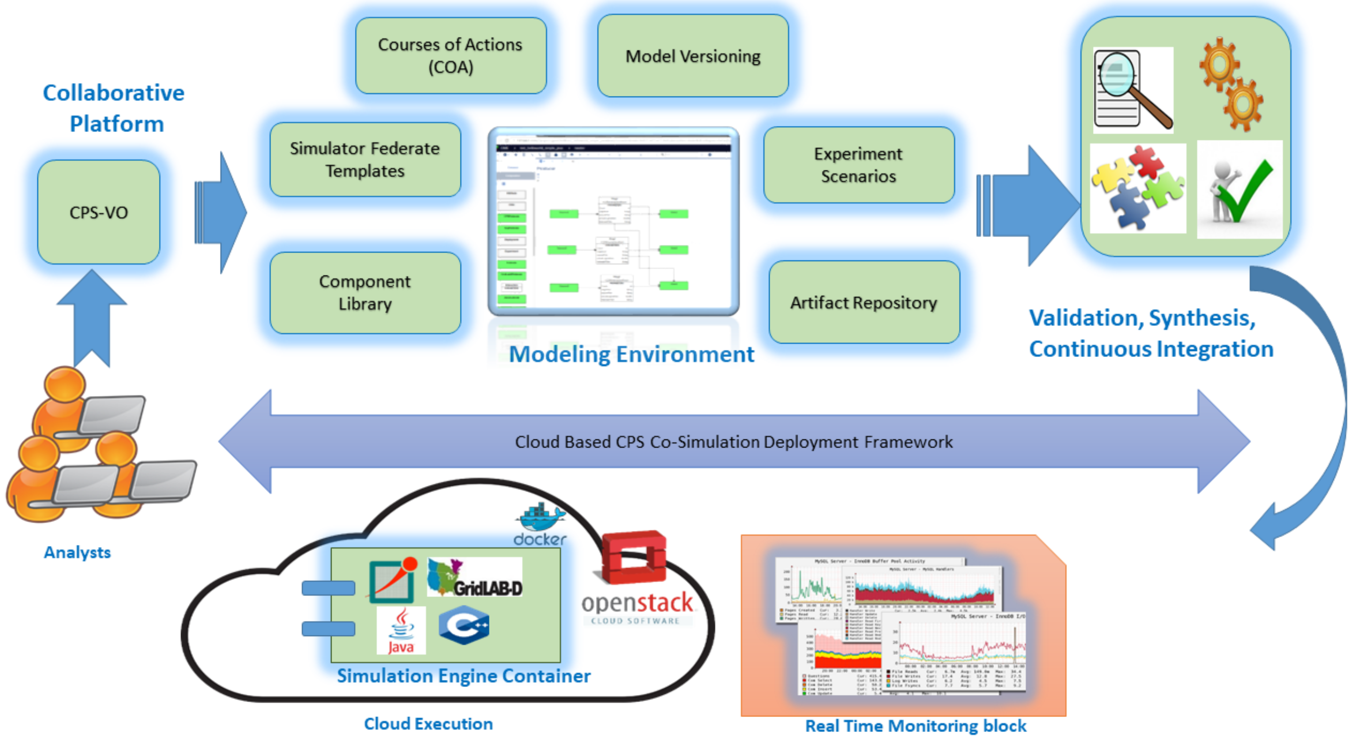

Fig. 2: Overview of the Cloud based Modeling and Co-simulation of Mixed Electrical Energy Systems

simulation scenario we could have more than one experiment model.

5) Software Synthesis: To facilitate rapid development of the co-simulation application, design studio features a code generation and synthesis module. This module enables synthesis of the *integration code* – software modules that bridge target simulators with the underlying HLA RTI. This module provides two key benefits. Firstly, it ensures the 'correct by construction' principle when generating the large boilerplate code required for such complex co-simulation RTI, thereby avoiding errors that occur with manually written code. Secondly, it lowers barrier to entry to the development of co-simulation application, whereby the domain expert can focus on writing simulation models and not worry about the complexity to work with the underlying RTI.

Apart from above software synthesis components, code generators also produce various experiment specific configurations, which are specific to the experiment model discussed earlier. These includes COA models, various federate configuration specific to the experiment model using JSON [16] files, and several initialization scripts.

6) Artifact Repository: All the artifacts that are part of the co-simulations are stored in a secured centralized repository with user access control. These artifacts includes autogenerated software codes, user supplied application programs, simulator specific models, WebGME models, and various configuration files associated with the co-simulation experiment. Artifact repository provides storage facility to store these co-simulation artifacts. The repository also features version control mechanism, such that the artifacts can be associated for a given version number and can optionally include labels such as development, beta, and production ready tags.

7) Continuous Integration: Continuous integration is widely used technique in the software development process for creating automated software builds that run various tests that validate whether the software compiles and builds successfully. It can also include unit tests to ensure that the newly developed simulation module meets certain functional requirements as specified in the unit tests. Continuous integration enables automatic compilation and building of the source code generated by the software synthesis module and storage of the compiled artifacts to the artifact repository discussed earlier. Currently, we are leveraging the Jenkins [17] build systems to trigger automatic builds of the simulation software.

8) Experimentation Runtime: To support large-scale distributed simulation, which may exhibit different compute, input/output, and/or network intensive workload characteristics, a cloud environment provides a better execution infrastructure to support these requirements. One of the requirements to support cross-platform executions is to enable running simulations across heterogeneous run-time platforms. Docker container technology [14] is utilized to meet this requirement. To enable running simulators inside docker containers, we first have to port the simulators to the docker run-time image. Once this image is created, it is available to run simulator-specific models. Currently, our design studio makes the C++, Java,

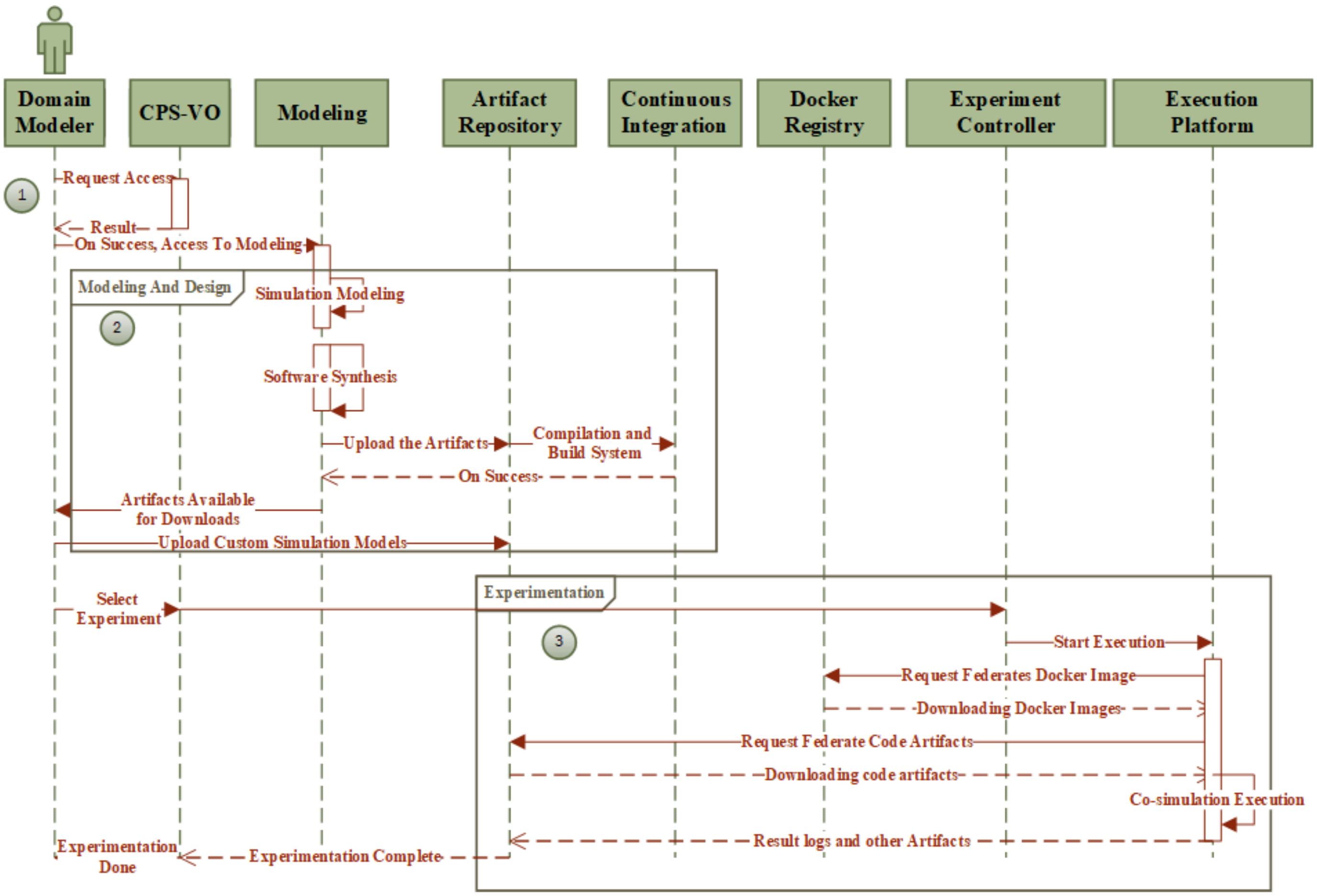

Fig. 3: Sequence diagram showcasing modeling and experimentation activity in design studio

OMNeT++, and CPNTools docker images available for the domain modelers.

9) Monitoring and Visualization: Live monitoring of co-simulation experiments provides insights into how the simulation experiment is running in the cloud environment. We are leveraging Grafana [18]) based visualization dashboards to depict important events that arise in the running simulation. Grafana dashboards also show the runtime system resource utilization of the individual federates comprising a co-simulation experiment run. Monitoring facility can thus be utilized for profiling simulation runs, and could be used in the future for making dynamic resource management decisions for running the distributed co-simulation experiments.

B. Modeling and Experimentation Workflow

In this section we will cover the modeling and experimentation workflow in the design studio framework. Figure 3 shows the two activities which the domain modeler performs in the design studio framework: co-simulation modeling and running co-simulations. Next, we will refer to the circle numbers as they are shown in the sequence diagram. In 1, the domain modeler first enters the design studio by authenticating with the CPS-VO portal. Once authenticated, the domain modeler has access to the WebGME modeling application. Next in step 2, the domain expert can first build the co-simulation design model and configure various experiment scenarios. Once the modeling activity is completed, software synthesis module generates simulation specific code artifacts. These artifacts are available to the modeler to download and update them with application-specific code and/or models. In addition, the software artifacts generated are passed to the continuous build and integration system for running versioned builds. The modeler can also upload custom simulation artifacts and updated software artifacts to the artifact repository, which can then be utilized accordingly for the co-simulation. Further, in step 3, the user than selects one of the configured experiment model from the previous step for executing on the experimentation platform. Once the run experimentation option is selected, the experiment controller then selects an appropriate runtime platform server from the available cloud infrastructure, and starts the experimental run sequence. During the startup of the experiment, appropriate Docker federate images, as required by the experiment, are downloaded from the docker registry hosted within the CPS-VO environment. Furthermore, the experiment specific federate code artifacts are downloaded on the execution server from the artifact repository. Once the required dependencies are downloaded, the co-simulation execution can begin. The simulation proceeds and simulates the experiment scenarios. Once the simulation criteria is met, which is set by the user based on either the amount of simulation time to execute or a specific simulation objective is met, the simulation execution stops. The simulation gets the execution trace and the generated results and logs are then uploaded back to the CPS-VO for the offline analysis.

IV. Conclusions And Future Work

This paper presents a cloud based secure, collaborative modeling and co-simulation platform to support mixed electrical energy systems simulation for smart grid operations. We have described various building blocks of the integrated design studio. We leverage the WebGME tool to provide a the web-based modeling environment and the CPS-VO to provide the central collaboration platform. Using continuous integration technology the simulation artifacts are automatically built and ready to be deployed to the cloud execution platform. Docker technology provides a cross-platform sandboxed execution environment for the co-simulations which can be executed on the elastic cloud computing resource.

There is a general lack of integrated toolsuites that can provide collaborative modeling and co-simulations facilities for complex applications such as mixed energy electrical systems. The presented design studio can enable various stakeholders in the smart grid environment to effectively collaborate with multi-user modeling of experiments, and to design and build resilient and high-performance smart grid systems. In future, we plan on adding support for efficient deployment and configuration management for the distributed simulations in the cloud environment [19], save and restore of the running simulations, and tool support for wider range of simulators such as EnergyPlus [20], and DIgSILENT [21].

Acknowledgments

This work was supported in part by National Science Foundation (NSF) under Award CNS-1521617. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

References

[1] "IEEE Std 15162010, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)- Framework and Rules," pp. 1–38, 2010.

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

[3] "Portico," Feb. 2018. [Online]. Available: https://github.com/openlvc/portico

[4] "Cyber-Physical Systems Virtual Organization (CPS-VO)," Feb. 2018. [Online]. Available: https://cps-vo.org

[5] M. Maróti, R. Kereskényi, T. Kecskés, P. Völgyesi, and A. Lédeczi, "Online collaborative environment for designing complex computational systems," *Procedia Computer Science*, vol. 29, pp. 2432–2441, 2014.

[6] D. C. Schmidt, "Model-driven engineering," *COMPUTER-IEEE COMPUTER SOCIETY-*, vol. 39, no. 2, p. 25, 2006.

[7] Y. D. Barve, P. Patil, and A. Gokhale, "A cloud-based immersive learning environment for distributed systems algorithms," in *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, vol. 1. IEEE, 2016, pp. 754–763.

[8] H. Neema, J. Sztipanovits, M. Burns, and E. Griffor, "C2WT-TE: A model-based open platform for integrated simulations of transactive smart grids," in *Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES), 2016 Workshop on*. IEEE, 2016, pp. 1–6.

[9] X. KoutsouKos, G. Karsai, A. Laszka, H. Neema, B. Potteiger, P. Volgyesi, Y. Vorobeychik, and J. Sztipanovits, "SURE: A modeling and simulation integration platform for evaluation of secure and resilient cyber–physical systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 93–112, 2018.

[10] Y. Barve, P. Patil, A. Bhattacharjee, and A. Gokhale, "Pads: Design and implementation of a cloud-based, immersive learning environment for distributed systems algorithms," *IEEE Transactions on Emerging Topics in Computing*, 2017.

[11] A. Bhattacharjee, Y. D. Barve, T. Kuroda, and A. Gokhale, "Cloudcamp: A model-driven generative approach for automating cloud application deployment and management," Institute for Software Integrated Systems, Vanderbilt University, Nashville, Report, 2017.

[12] S. Shekhar, Y. Barve, and A. Gokhale, "Understanding performance interference benchmarking and application profiling techniques for cloud-hosted latency-sensitive applications," in *Proceedings of the10th International Conference on Utility and Cloud Computing*, ser. UCC '17. New York, NY, USA: ACM, 2017, pp. 187–188. [Online]. Available: http://doi.acm.org/10.1145/3147213.3149453

[13] "OpenStack," Feb. 2018. [Online]. Available: https://www.openstack.org

[14] "Docker," Feb. 2018. [Online]. Available: https://www.docker.com

[15] H. Neema, G. Karsai, and A. H. Levis, "Next-generation command and control wind tunnel for courses of action simulation," Institute for Software-Integrated Systems, Vanderbilt University, Tech. Rep. no. ISIS-15-119, 2015.

[16] "JavaScript Object Notation (JSON)," Feb. 2018. [Online]. Available: https://www.json.org

[17] "Jenkins," Feb. 2018. [Online]. Available: https://www.jenkins.io

[18] "Grafana," Feb. 2018. [Online]. Available: https://www.grafana.com

[19] Y. D. Barve, H. Neema, A. Gokhale, and S. Janos, "Model-driven automated deployment of large-scale cps co-simulations in the cloud (poster)," in *ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, Austin, TX, 09/2017 2017.

[20] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte *et al.*, "Energyplus: creating a new-generation building energy simulation program," *Energy and buildings*, vol. 33, no. 4, pp. 319–331, 2001.

[21] A. D. Hansen, C. Jauch, P. Sørensen, F. Iov, and F. Blaabjerg, "Dynamic wind turbine models in power system simulation tool digsilent," *Report Risoe*, pp. 1–80, 2003.