Model-driven Automated Deployment of Large-scale CPS Co-simulations in the Cloud

Yogesh D. Barve, Himanshu Neema, Aniruddha Gokhale and Janos Sztipanovits
Institute for Software-Integrated Systems,
Dept. of EECS, Vanderbilt University,
Nashville, TN 37212, USA
Email:{yogesh.d.barve,himanshu.neema, a.gokhale, janos.sztipanovits}@vanderbilt.edu

Abstract—**With increasing advances in Internet-enabled devices, large cyber-physical systems (CPS) are being realized by integrating several sub-systems together. Analyzing and reasoning different properties of such CPS requires co-simulations by composing individual and heterogeneous simulators, each of which addresses only certain aspects of the CPS. Often these co-simulations are realized as point solutions or composed in an ad hoc manner, which makes it hard to reuse, maintain and evolve these co-simulations. Although our prior work on a model-based framework called Command and Control Wind Tunnel (C2WT) supports distributed co-simulations, many challenges remain unresolved. For instance, evaluating these complex CPSs requires large amount of computational and I/O resources for which the cloud is an attractive option yet there is a general lack of scientific approaches to deploy co-simulations in the cloud. In this context, the key challenges include (i) rapid provisioning and de-provisioning of experimental resources in the cloud for different co-simulation workloads, (ii) simulating incompatibility and resource violations, (iii) reliable execution of co-simulation experiments, and (iv) reproducible experiments. Our solution builds upon the C2WT heterogeneous simulation integration technology and leverages the Docker container technology to provide a model-driven integrated tool-suite for specifying experiment and resource requirements, and deploying repeatable cloud-scale experiments. In this work, we present the core concepts and architecture of our framework, and provide a summary of our current work in addressing these challenges.**

Index Terms—**co-simulations, verification, model driven, cloud**

I. INTRODUCTION AND PROBLEM STATEMENT

Large-scale cyber physical systems (CPS) experiments are being increasingly deployed for real-world scenarios in domains such as building automation and control, smart power grid, health-care, and industrial processes. For example, power grid CPSs are composed of many multi-domain subsystems with different assets and technologies, such as electric grid, sensors, networking and physical control systems. Thus, designing and analyzing such complex systems needs extensive simulation and prototyping tools that span multiple domains.

While recent advances in simulation tools have enabled modeling and simulation of system characteristics, a single simulator tool is not sufficient to model and experiment with CPS. This is due to the fact that no single simulator can simulate all aspects of CPS, and moreover, CPS require heterogeneous resources and execution environments. Thus co-simulation environments have emerged as an approach for modeling and simulating CPS. Co-simulation or coupled simulation is a methodology that focuses on evaluating the behavior of a system by integrating simulations of its components. Each specialized simulation tool can process and communicate various events among participating simulation engines to model large-scale CPS. To realize such a co-simulation platform, proper time synchronization and coordination of message flows among participating simulations engines is needed.

C2WT [1] is a heterogeneous simulation integration framework that we have previously developed at Vanderbilt University. It enables model-based rapid synthesis of heterogeneous and distributed CPS co-simulations. C2WT relies on the IEEE High-Level Architecture (HLA) standard. Domain-specific tools have been built on top of C2WT such as C2WT-TE [2] which targets transactive smart grid domain, and the SURE testbed [3] that targets security and resilience in CPS.

Despite these advances, many challenges still remain unresolved. For instance, large-scale simulations exhibit compute and/or I/O intensive workloads and may need large amount of such resources. Cloud computing can provide access to such a large pool of resources elastically and on-demand. However, existing cloud platforms lack tools for effective deployment of large-scale CPS simulations. Migrating existing simulation tools to the cloud is also a challenging task, which hinders the widespread adoption of cloud computing for CPS co-simulation. This problem is further exacerbated since CPS domain experts conducting the simulations often lack a proper understanding of the cloud resource provisioning and utilization thereby resulting in ad hoc and sub-optimal deployment of CPS simulations in the cloud.

In this research, we focus primarily on cloud-based provisioning of large-scale CPS experiments, and outline the key challenges associated with deploying and experimenting with CPS co-simulations in the cloud.

II. CHALLENGES IN REALIZING CLOUD-HOSTED CPS CO-SIMULATIONS

The following challenges must be resolved to support reusable and extensible cloud-based CPS co-simulations.

1. Integrated tool to rapidly deploy experiments on cloud resources: To run experiments in the cloud, the framework should be able to acquire required resources, instantiate the deployment and execution of the co-simulation, and tear down

the acquired resources when the experiment is completed. The run-time infrastructure should require minimal startup and shutdown time to ensure a quick experiment start and prompt release of resources, without incurring additional resource utilization cost. The simulations also impose different resource requirements such as CPU cores, GPU, RAM, and disk space. Moreover, the simulations could be CPU and/or I/O intensive. These resource requirements must be configured in the tool, and the cloud resources should be allocated accordingly. A dynamic cloud resource management strategy can be highly effective for better cloud resource utilization.

2. Handling simulation incompatibility and resource violations: For faithful experimental outcomes, different simulators impose co-simulation specific data-exchange requirements and QoS constraints such as communication latencies, computation execution deadlines, hardware resource availability (CPUs, memory, etc.). For instance, if one of the simulators in the co-simulation requires high I/O bandwidth to stream large videos, the receiving simulator needs to consume the streamed data within a given time-period. Thus, if these simulators are not co-located in the cloud, a violation or incompatibility warning should be raised so that the user can make the necessary modifications to satisfy the QoS constraints.

3. Proactive fault tolerance for simulation execution: The cloud-based co-simulation framework must be resilient to system faults and failures that can occur within the cloud platforms. Our solution, called *Co-simulation checkpointing*, leverages Linux container's save and restore functions, and enables, in the event of a failure, an effective recovery of systems to their previously checkpointed states. Implementing checkpoint with distributed co-simulations, that have intertwined dependencies, is even more challenging. Here, checkpointing also needs to be coordinated and synchronized across all simulators. This ensures reliable recovery and correct execution from snapshot images during system restoration.

4. Reproducible Experiments: Deterministic execution and reproducible experiments are needed for many CPS co-simulations. The co-simulation integration methods and runtime execution tools must be designed for these requirements from the start. In addition, for repeatable experiments, the cloud experimentation platform should provide the same runtime execution environment and configuration for the same experiments.

III. PROPOSED SOLUTION AND CURRENT STATUS

We are developing a framework that enables effective CPS co-simulation in cloud. Figure 1 provides the functional architecture of our framework. Our framework uses Docker containers for deploying simulations in an Openstack cloud. The simulations are built using corresponding pre-packaged simulators inside a Docker container. These containers provide a repeatable runtime environment. We are also developing a domain-specific modeling language to capture experiment resource requirements of individual simulators.

In future, we plan on integrating an SMT solver for an optimal placement of simulators in the cloud environment
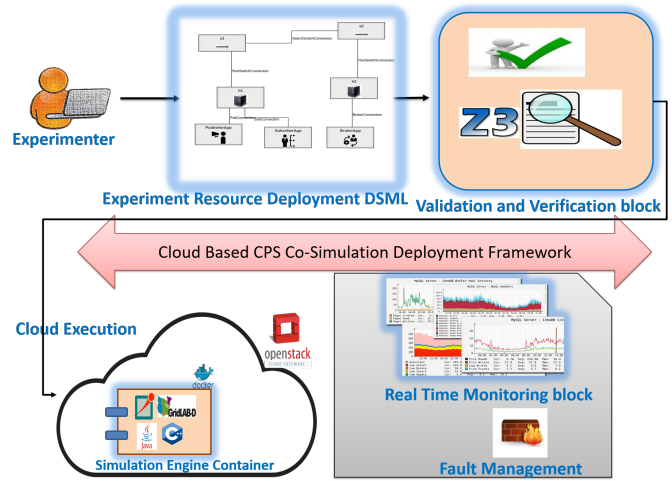

Fig. 1: Architecture Overview of CPS Co-simulation Deployment in the Cloud

while still satisfying individual resource requirements. We are also building a cloud resource monitoring framework utilizing collectd and other tools to enable real time monitoring of cloud resources which can then be fed to the SMT solvers to make effective decisions. To enable fault-tolerant co-simulations, we are developing a *Co-simulation checkpointing* technique using the save and restore functions of the CRIU library for Docker containers. It is also critical that checkpointing should be synchronized and coordinated, and must support distributed simulator deployments.

REFERENCES

[1] 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.

[2] 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.

[3] H. Neema, P. Volgyesi, B. Potteiger, W. Emfinger, X. Koutsoukos, G. Karsai, Y. Vorobeychik, and J. Sztipanovits, "SURE: An Experimentation and Evaluation Testbed for CPS Security and Resilience: Demo Abstract," in *Proceedings of the 7th International Conference on Cyber-Physical Systems*, ser. ICCPS '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 27:1–27:1. [Online]. Available: http://dl.acm.org/citation.cfm?id=2984464.2984491