The Journal of Open Source Software

CoSim Toolbox - Making HELICS Co-Simulations Easier

Trevor Hardy [1*], **Mitch Pelton**[1*], **Nathan Gray** [1*], **Fred Rutz** [1], and **Wen Bo** [1]

1 Pacific Northwest National Laboratory * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 03 October 2025
Published: unpublished

Summary

Co-simulation is a technique for creating larger and more complex time-series system models by facilitating data exchange between two or more models during runtime. This data exchange allows boundary states in models which may have been defined through static values or historical values to be dynamically generated by models in other simulator instances. This dynamic data exchange allows for system models that transcend individual simulation tools, modeled domains, and modeled subsystems. FMI (FMI, 2025), particularly as implemented in Modelon, is an example of a popular co-simulation platform.

HELICS (Hardy et al., 2024) is the US Department of Energy's publicly available co-simulation platform and has been under development for over eight years. HELICS provides the necessary simulation time management and facilitates the data exhanges between simulator instances (called "federates") and has been sucessfully used in a number of published studies such as (Theisen et al., 2024), (Bharati et al., 2023), (Panossian et al., 2023) and (Lardier, 2020). Though these studies largely focus on power system applications, HELICS itself is domain agnostic.

Statement of Need

CoSim Toolbox (CST) is built on top of HELICS and has the goal of making HELICS-based co-simulations easier to manage. Based on years of experience using HELICS, we have found a number of pain points around data and metadata management and HELICS federate creation that CoSim Toolbox is designed to alleviate. Specifically, we have integrated the following tools and code to make using HELICS easier.

- **Docker** - Docker containers are used to distribute HELICS, the CST API as well as any CST-enabled simulation tools. This allows the use of CST on all Linux, MacOS, and Windows while minimizing the maintenance of these codebases.
- **Generic Federate class** - To facilitate writing new federates more easily, CST provides a generic federate class for use when writing new Python federates. This class wraps the HELICS APIs and facilitates data exchanges using just Python data structures. By minimizing the amount of knowledge of HELICS APIs, we have seen those with no knowledge of HELICS be able to more easily implement a HELICS-based co-simulation using this CST federate class.
- **Time-Series Data Collection** - A Postgres database is used to collect time-series data being generated by every federate. A data collection federate has been written using the CST APIs to collect data generated by the HELICS-enabled data exchanges and automatically commit it to the Postgres database. This allows data to be stored and thus accessed in a central location, making it easier to monitor and post-process the

co-simulation data. The Postgres database is implemented as a Docker container along with pgAdmin to provide a web interface for inspecting the Postgres database.

- **Metadata and Configuration Management** - HELICS federates require significant configuration which has historically been accomplished by JSON files on disk. Particuarly for larger federations, this leads to a significant number of JSON files, sometimes leading to confusion about which files correspond to the data produced in a given co-simulation run. To help manage this configuration data, CST uses MongoDB to store and the configuration information with the CST federate class designed to automatically retrieve a given federates configuration. The MongoDB database is implemented as a Docker container along with Mongo Express to provide a a web interface for inspecting the MongoDB database.

Acknowlegements

This work was funded under the E-COMP, a laboratory-directed research and development initiative at the Pacific Northwest National Laboratory, part of the United States Department of Energy.

References

Bharati, A. K., Ajjarapu, V., Du, W., & Liu, Y. (2023). Role of distributed inverter-based-resources in bulk grid primary frequency response through HELICS based SMTD co-simulation. *IEEE Systems Journal*, *17*(1), 1071–1082. https://doi.org/%7B10.1109/JSYST.2022.3218117%7D

FMI, M. A. P. (2025). Fmi: Functional mock-up interface. In *GitHub repository*. GitHub. https://github.com/modelica/fmi-standard

Hardy, T. D., Palmintier, B., Top, P. L., Krishnamurthy, D., & Fuller, J. C. (2024). HELICS: A Co-Simulation Framework for Scalable Multi-Domain Modeling and Analysis. *IEEE Access*, *12*, 24325–24347. https://doi.org/%7B10.1109/ACCESS.2024.3363615%7D

Lardier, W. (2020). *ASGARDS-h: Enabling advanced smart grid cyber-physical attacks, risk and data studies with HELICS* [Master's thesis, Concordia University]. https://spectrum.library.concordia.ca/id/eprint/987690/1/Lardier_MASc_S2021.pdf

Panossian, N. V., Laarabi, H., Moffat, K., Chang, H., Palmintier, B., Meintz, A., Lipman, T. E., & Waraich, R. A. (2023). Architecture for co-simulation of transportation and distribution systems with electric vehicle charging at scale in the san francisco bay area. *Energies*, *16*(5). https://doi.org/%7B10.3390/en16052189%7D

Theisen, J. R., Bose, A., Mukherjee, M., Burgess, D., Wilhelm, K., & Diedesch, M. (2024). Community-based transactive coordination mechanism for enabling grid-edge systems. *2024 IEEE Texas Power and Energy Conference (TPEC)*, 1–6. https://doi.org/%7B10.1109/TPEC60005.2024.10472237%7D