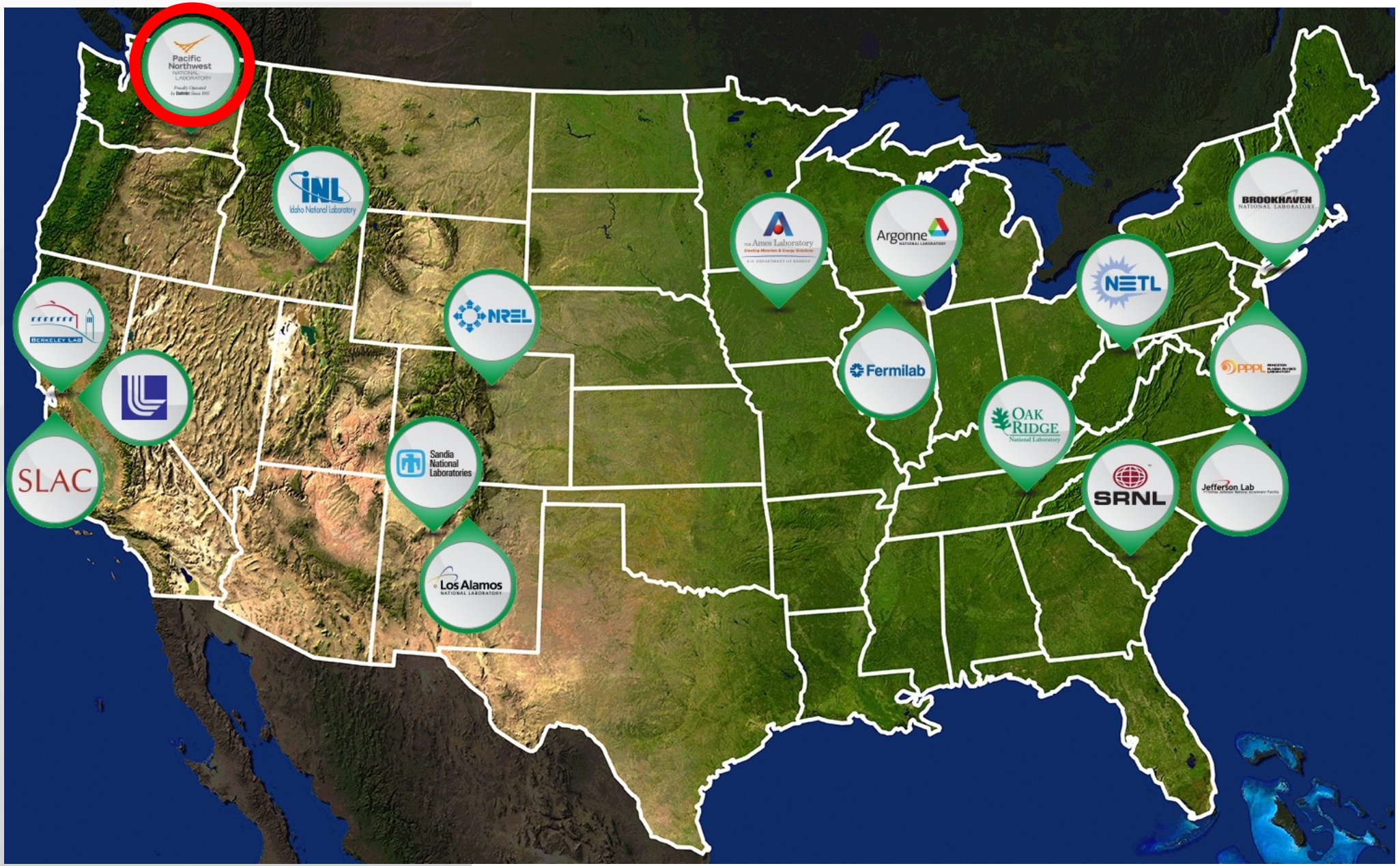
# Co-Simulation as a Technique for Large-Scale and Complex Multi-Energy Modeling

**Dr. Trevor Hardy**

**Power Systems Research Engineer**
**Pacific Northwest National Laboratory**

**17 June 2021**

# United States Department of Energy National Laboratory System



# The Need for Co-Simulation – One Simulator is Not Enough

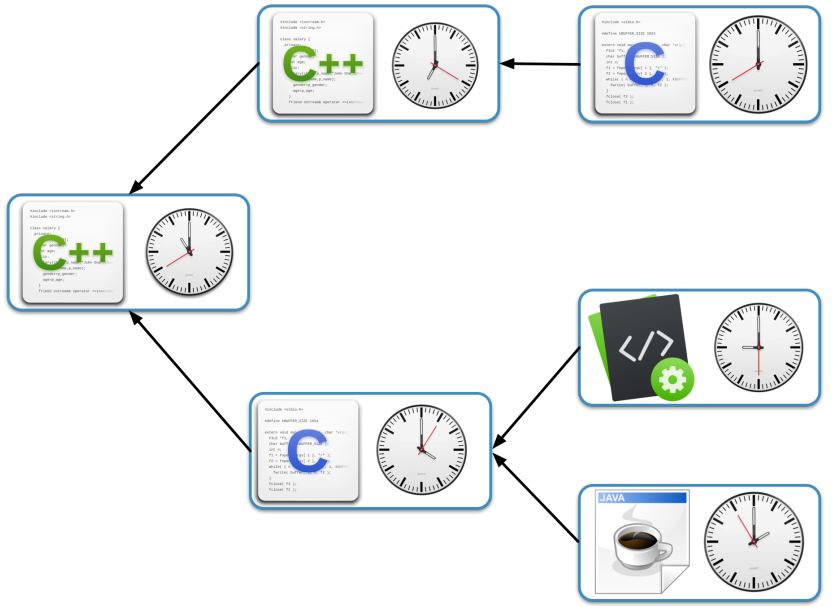
- Analysis requires the dynamic, two-way interaction between two or more simulation tools and their models
  - Recorded datasets as inputs are not good enough and dynamic interaction is required to meet analysis goals
- Analysis requires development of a system model that transcends traditional simulation tool boundaries
  - Buildings + Power Grid
  - Power Grid + Natural Gas
- Analysis requires a large-scale model not practically captured within a single simulation tool instance.
  - Texas power grid during winter cold snap

# Analyzing Large and Complex Systems with Co-Simulation

- Co-simulation is a method for unifying multiple models in a coordinated fashion. Models may or may not:
  - Cover similar domains
  - Have a similar concept of time
  - Be written in the same language
  - Be run by simulators on the same computer or operating system, or be in the same network
- Co-simulation allows models that have interactions with each other to express those interactions and influence each other's behavior.

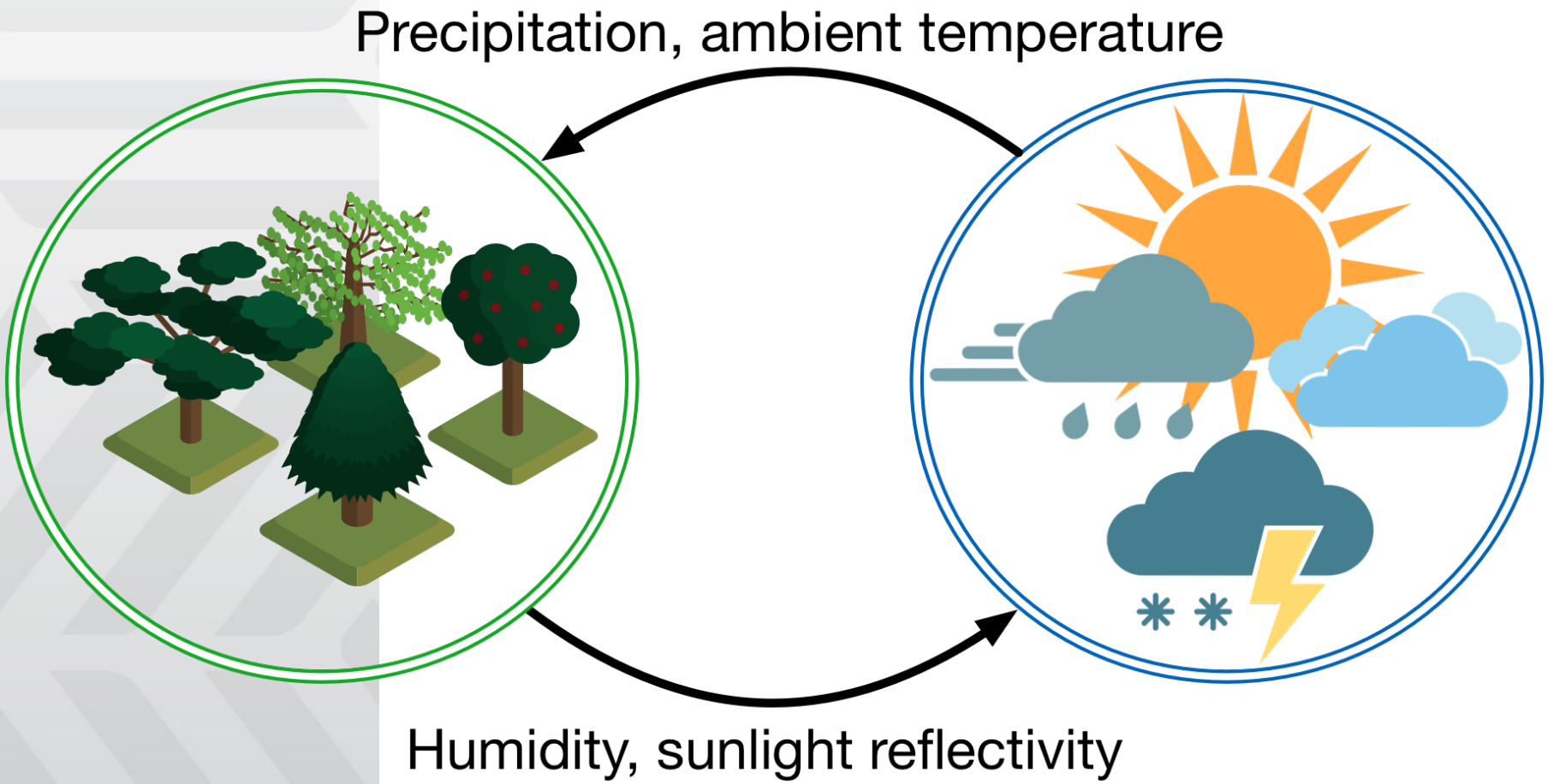
# Co-Simulation Platform Key Functions

- Simulator Time Synchronization




- Simulator Data Exchange
  - Timely distribution of appropriate data
  - Data type management
  - Structured data support (*e.g.* JSONs)
  - Data privacy

# Co-Simulation Example



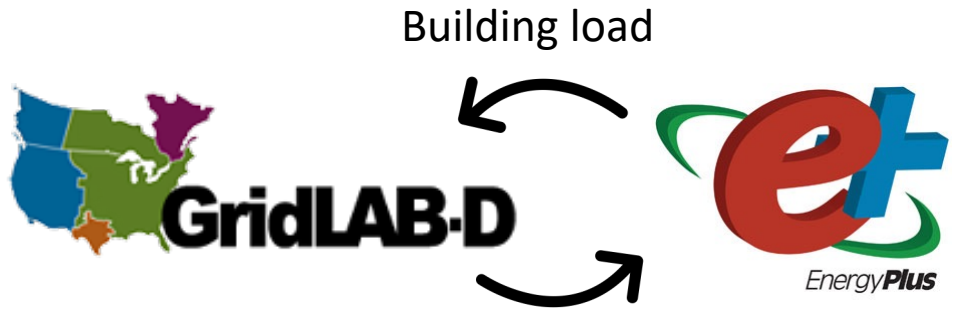
# HELICS: Multi-Lab DOE-Sponsored Co-Simulation Platform

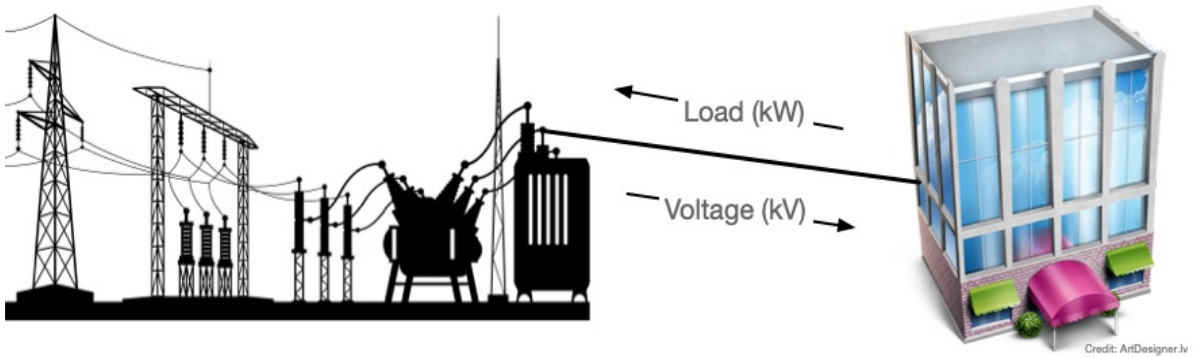

HELICS co-simulation platform is composed of:

- Libraries and language bindings to use for integrating a simulator
  - C
  - C++
  - Python
  - MATLAB
  - Julia
  - …
- Executables for running co-simulations and assisting in co-simulation development and testing
  - Data exchange and synchronization services

# Co-Simulation Interaction Types

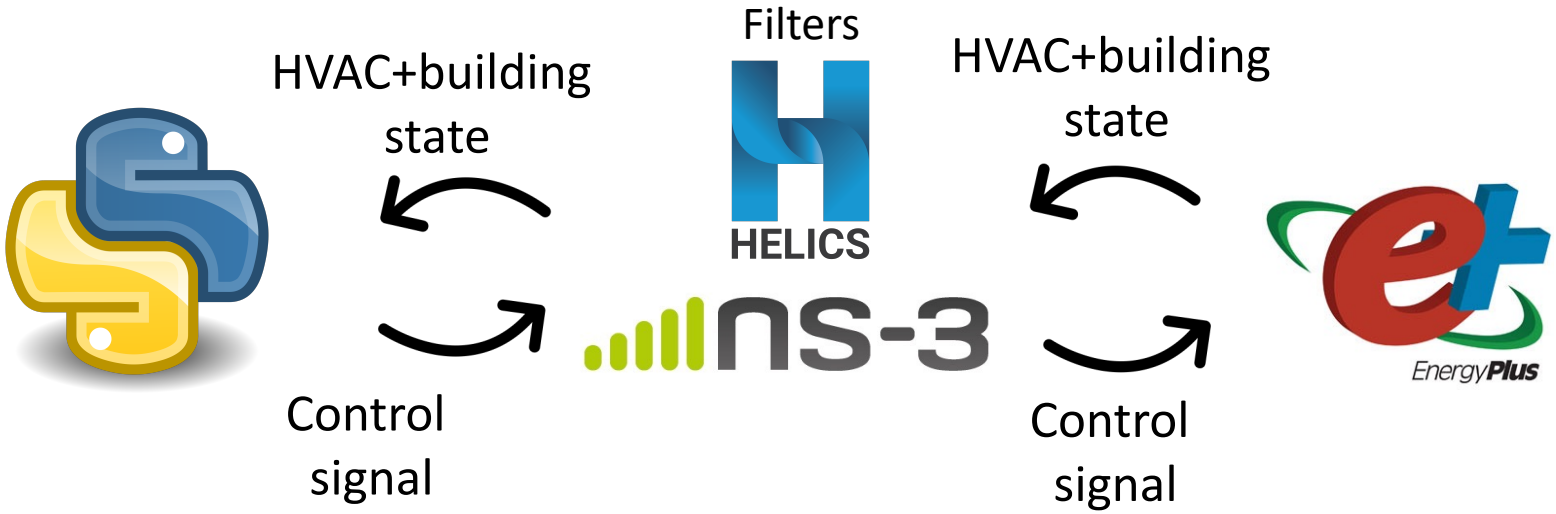
- Physics simulation: value exchanges

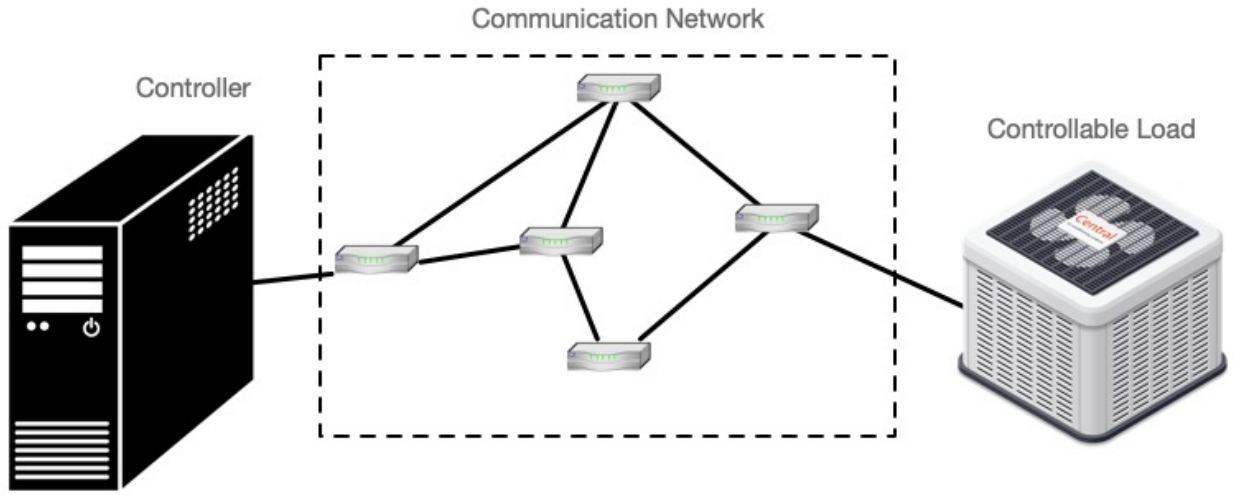


# Co-Simulation Interaction Types

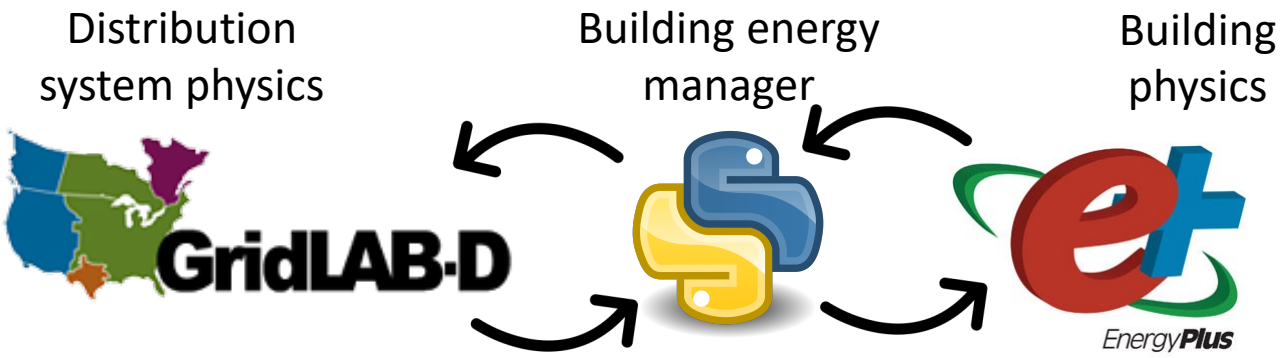
- Communication signals: message exchanges



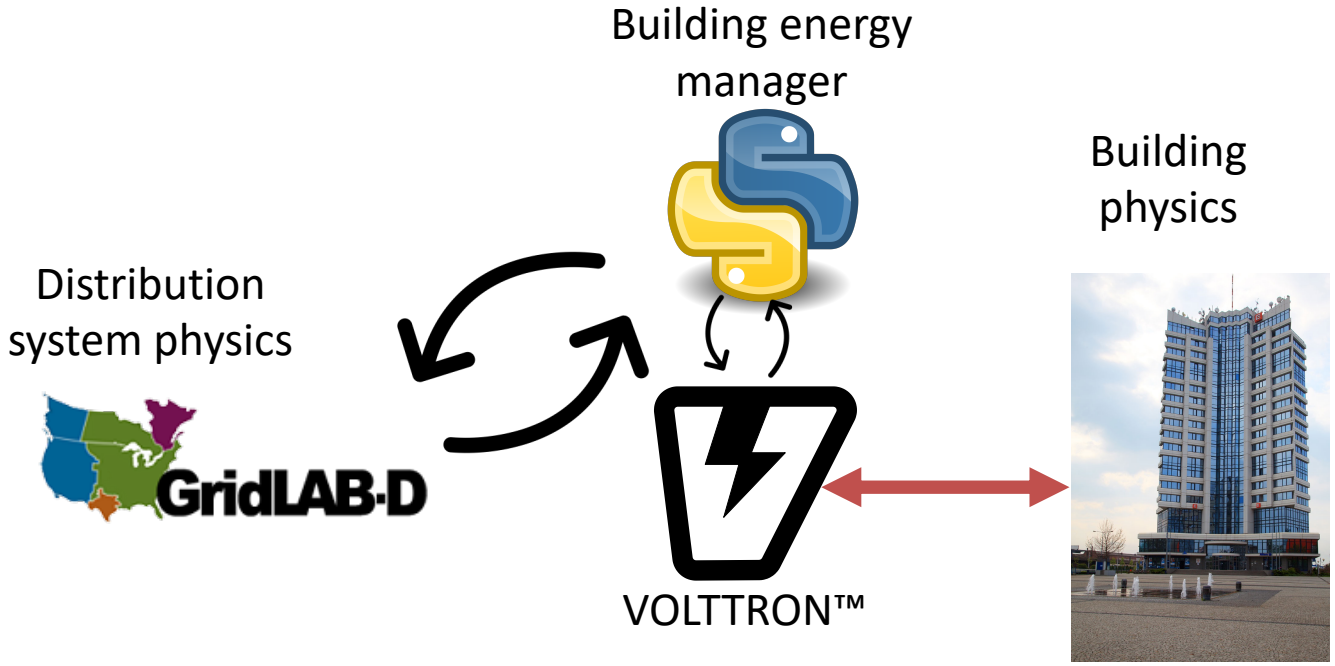



# Co-Simulation with Hardware

- Software-only co-simulation



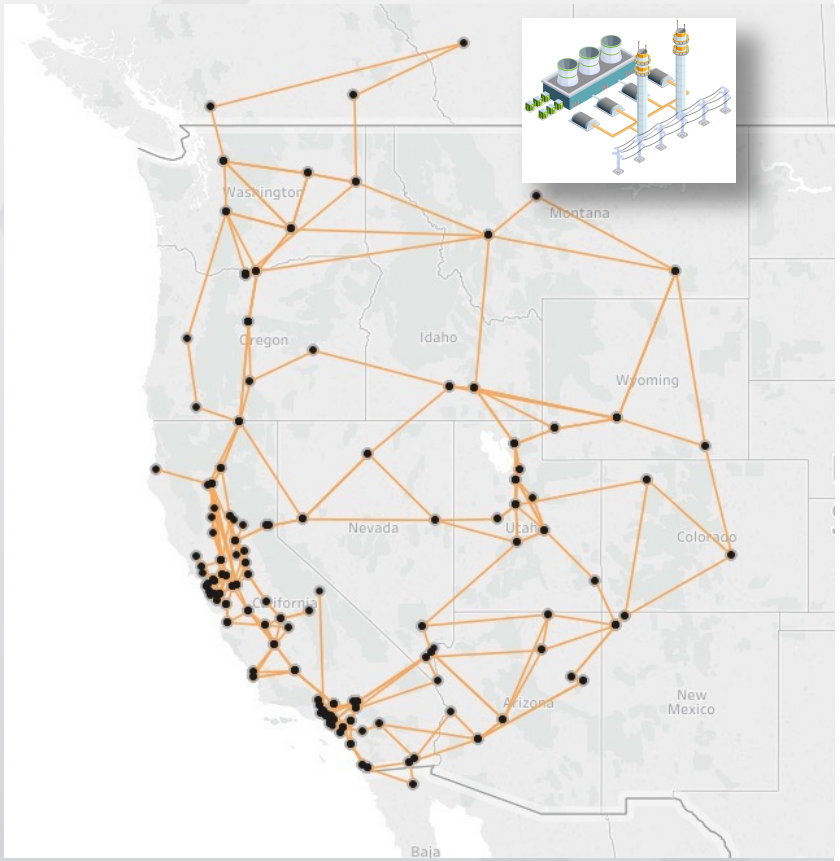

- Hardware-in-the-loop (HIL) co-simulation



# Co-Simulation for Multi-Energy Modeling

- Power System and Natural Gas

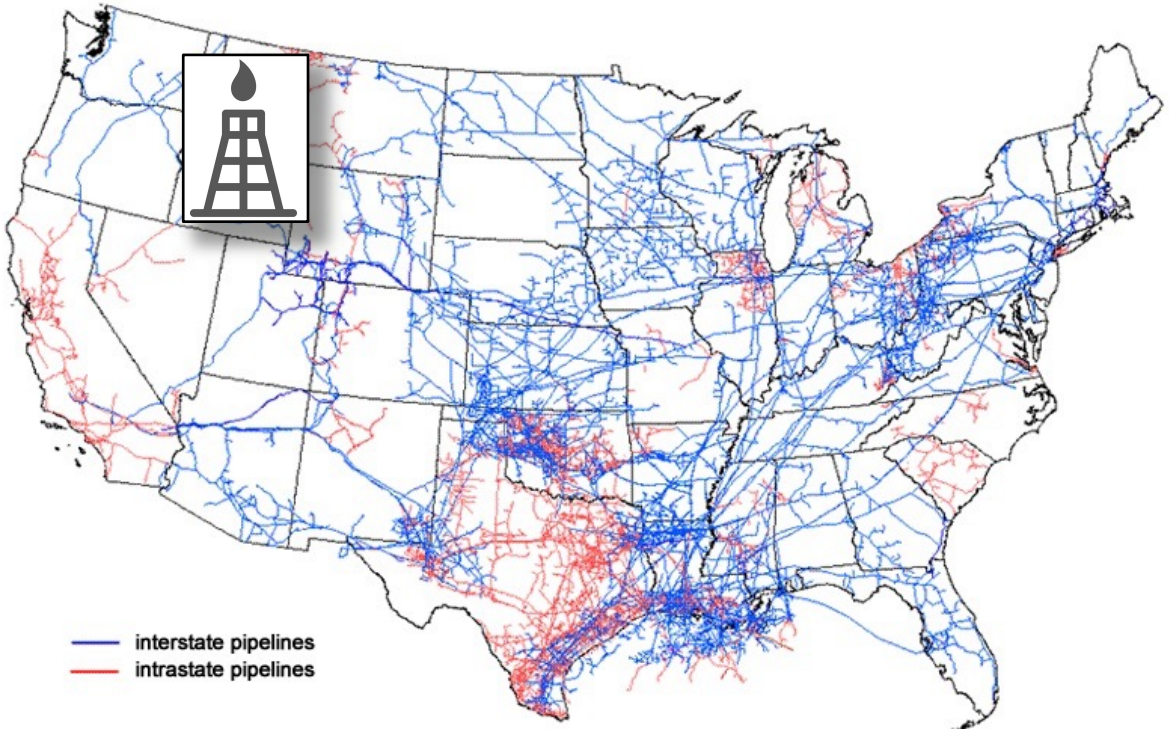
Power system physics and operations




Power plant gas pressure

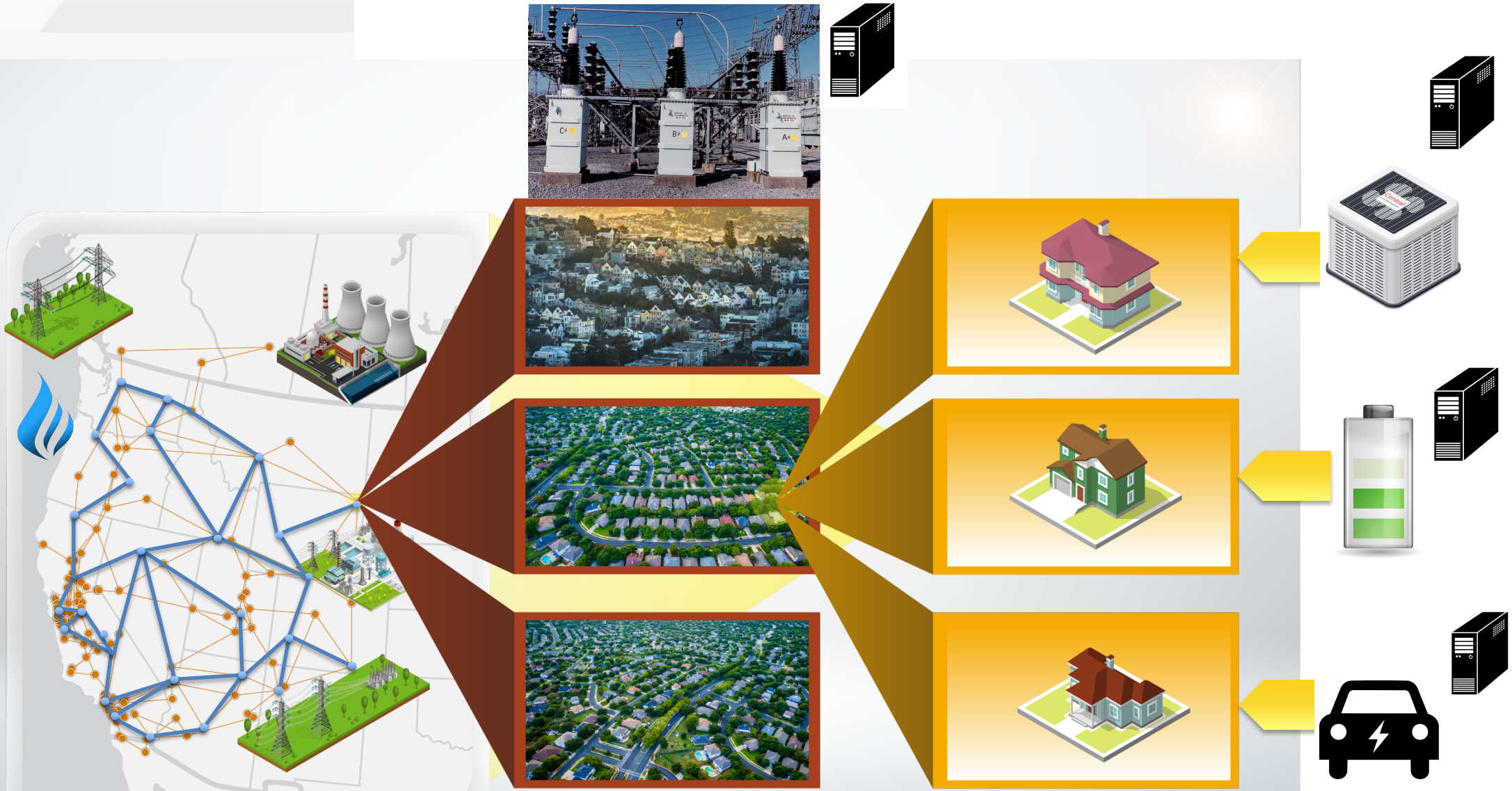
Gas plant electrical power

Natural gas system physics and operations



# Large-Scale Power System Example

# Large-Scale and Complex Energy System Example



# Popular HELICS Alternatives



- Functional Mock-up Interface (FMI)
  - Started out as a means of exchanging dynamic models for automotive components while protecting IP
  - Individual models are called “FMUs”
  - Models must be linked by a master algorithm
  - Most popular implementation is in/with Modelica
  - HELICS has an FMI interface and acts as the master algorithm

- mosaik
  - Generic co-simulation framework that started out with a power system focus, just like HELICS.
  - Python based
    - ✓ HELICS is C++-based with bindings for a variety of languages including Python
  - Communication over network sockets
    - ✓ HELICS also uses network sockets but implements a variety of messaging technologies to fit particular computing environments

trevor.hardy@pnnl.gov

https://www.helics.org

https://github.com/GMLC-TDC/

https://helics.readthedocs.io

https://helics.readthedocs.io/en/latest/user-guide/index.html