



## Control of Complex Systems: An Integrated Perspective on Modern Power Grid Control

# Arion: High Level Modeling Language for Developing Co-simulations of Complex Systems

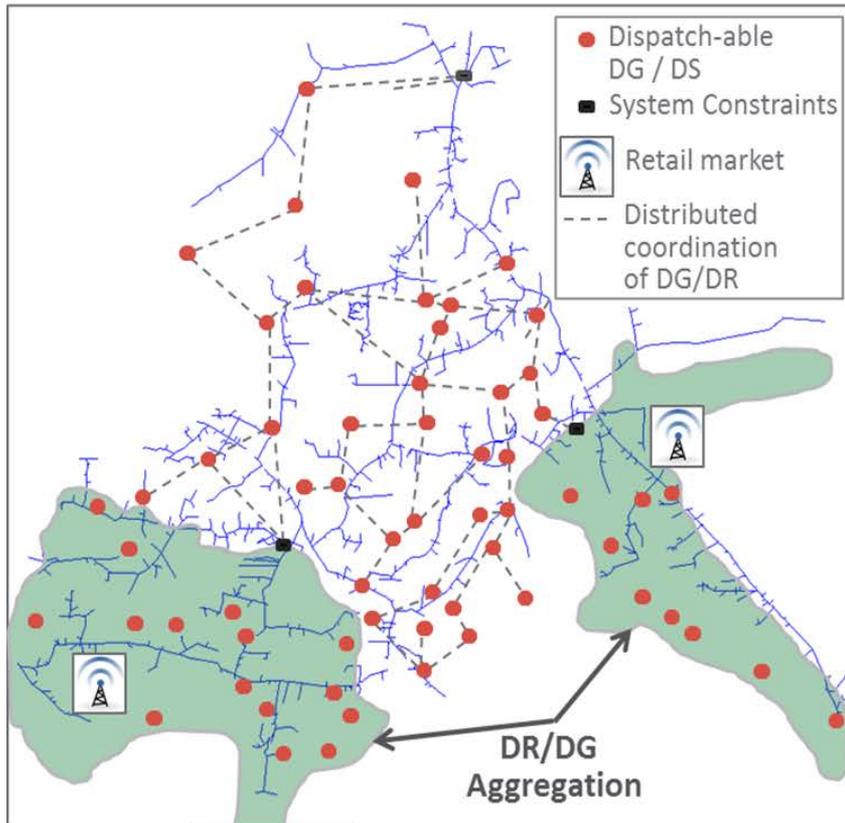
THOMAS W. EDGAR

Pacific Northwest National Laboratory

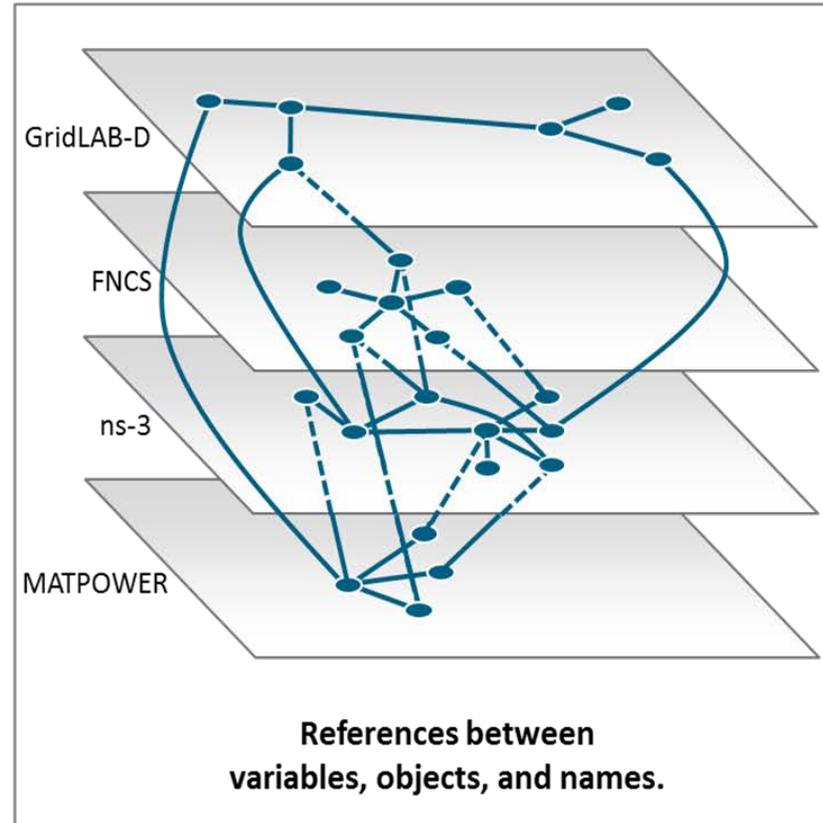


# Modeling Complex Systems

## Researcher View of Model



## Technology View of Model



Need: Consistent, integrated modeling language and translators



# High Level Modeling Language (Arion)

- Enable researchers to define complex system models holistically
  - Provide abstractions of test bed capabilities
  - No need to worry about technology configuration
- Leveraged the Java language and ecosystem to develop modeling language
- Used reflection and dynamic compilation to obscure compiled language
- Built libraries of objects and models across simulators
- Merged concepts across simulators for cohesive modeling language





# Arion Code Translation

## Arion File

```
// Create a transformer configuration for the substation
final TransformerConfiguration substationConfig = sim.transformerConfiguration("substation_config");
substationConfig.setConnectionType(ConnectionType.AVE_AWV);
substationConfig.setInstallationType(InstallationType.PADMOUNT);
substationConfig.setPrimaryVoltage(7200);
substationConfig.setSecondaryVoltage(7200);
substationConfig.setPowerRating(3 * numHouses * 5.0);
substationConfig.setPhaseRating(numHouses * 5.0);
substationConfig.setPhaseKrating(numHouses * 5.0);
substationConfig.setImpedance(0.0015, 0.00675);

// Create the transformer configuration for each phase from the substation
final TransformerConfiguration defaultTransformerA = sim.transformerConfiguration("default_transformer_A");
defaultTransformerA.setPhaseRating(numHouses * 5.0);
defaultTransformerA.setPhaseKrating(numHouses * 5.0);
defaultTransformerA.setImpedance(0.0015, 0.00675);

final TransformerConfiguration defaultTransformerB = sim.transformerConfiguration("default_transformer_B");
defaultTransformerB.setPhaseRating(numHouses * 5.0);
defaultTransformerB.setPhaseKrating(numHouses * 5.0);
defaultTransformerB.setImpedance(0.0015, 0.00675);

final TransformerConfiguration defaultTransformerC = sim.transformerConfiguration("default_transformer_C");
defaultTransformerC.setPhaseRating(numHouses * 5.0);
defaultTransformerC.setPhaseKrating(numHouses * 5.0);
defaultTransformerC.setImpedance(0.0015, 0.00675);

// Create the triplex line conductor used everywhere
final TriplexLineConductor triplexConductorA = sim.triplexLineConductor("Name_1_0_AA_triplex");
triplexConductorA.setResistance(0.57);
triplexConductorA.setGeometricMeanRadius(0.0111);

// Create the triplex line configuration used everywhere
final TriplexLineConfiguration triplexLineConf = sim.triplexLineConfiguration("TLCFG");
triplexLineConf.setPhaseConductor(triplexConductorA);
triplexLineConf.setPhaseConductor(triplexConductorA);
triplexLineConf.setPhaseConductor(triplexConductorA);
triplexLineConf.setInsulationThickness(0.005);
triplexLineConf.setDiameter(0.368);

// Create the base substation
final Substation substation = sim.substation("substation_root");
substation.setBusType(BusType.SMND);
substation.setBusInkVoltage(7200.0);
substation.setReferencePhase(PhaseCode.A);
substation.setPhases(PhaseCode.ABCN);
substation.setVoltage(7200.0, 0.0);
substation.setVoltage(-3600.0, -6235.3829);
substation.setVoltage(-3600.0, 6235.3829);
```

Infrastructure

Experimental Setup

Experimental Model



```
CREATE DATABASE conf_experiment;
CREATE TABLE Markets (
  Time DATE NOT NULL,
  clearing_quantity INTEGER,
  base_price INTEGER,
  current_price_upper_24h INTEGER,
  current_price_lower_24h INTEGER,
  request_for_bidding_bid_price INTEGER,
  current_market_clearing_price INTEGER,
  current_market_clearing_quantity INTEGER,
  adjusted_price INTEGER);
CREATE TABLE Substation (
  associated_real_power INTEGER,
  distribution_load_level INTEGER);
```

Conference Servers and Network

```
CREATE DATABASE conf_experiment;
CREATE TABLE Markets (
  Time DATE NOT NULL,
  clearing_quantity INTEGER,
  base_price INTEGER,
  current_price_upper_24h INTEGER,
  current_price_lower_24h INTEGER,
  request_for_bidding_bid_price INTEGER,
  current_market_clearing_price INTEGER,
  current_market_clearing_quantity INTEGER,
  adjusted_price INTEGER);
CREATE TABLE Substation (
  associated_real_power INTEGER,
  distribution_load_level INTEGER);
```

Configure Experimental Connections for Co-Simulation and Data Collection

```
{
  "interface": "EngnetworkInterface",
  "broker": "localhost",
  "simulator_type": "power_grid",
  "synchronization_algorithm": "GracePeriodAndSlip",
  "sync_ticks": 1,
  "number_of_power_grid_sims": 2,
  "simulator_time_unit": "seconds",
  "socket_timeout": 280000000
}
```

```
subject triplex_line {
  type TriplexLine;
  name "TLCFG";
  length 10;
  num_phases 3;
  subject triplex_conductor {
    name "TLCFG";
    length 10;
    num_phases 3;
  }
  subject triplex_conf {
    name "TLCFG";
    length 10;
    num_phases 3;
  }
}
```

Configure Simulations of Interest

```
subject substation {
  type Substation;
  name "substation_root";
  length 10;
  num_phases 3;
  subject transformer {
    name "substation_root";
    length 10;
    num_phases 3;
  }
}
```





# Benefits

## Arion

```

CsmaChannel channel = new CsmaChannel("F1_C_NI1", dataRate, delay);
channel.setAddressBase(ipBase);
addChannel(channel);

Node controller = new Node("F1_C_NI1");
NetDeviceContainer csmaDevices = new NetDeviceContainer("F1_C_NI1");

csmaHelper.install(controller, channel, csmaDevices);
iStackHelper.install(controller);

GldHouseUtils.generateHouse(sim, id, meter, triplineConf, auction, phase,
    controllerNIPrefix, channel, track, rand);

names.pushBack(controller);
marketToControllerMap.put(auction.getNetworkInterfaceName(),
    auction.getFnccsControllerPrefix());

```

```

for (int i = 1; i <= numHouses; i++) {
    TriplexMeter meter;
    PhaseCode phase;
    if (i <= numHouses / 3) {
        meter = triplexMeterA;
        phase = PhaseCode.A;
    } else if (i <= (numHouses * 2) / 3) {
        meter = triplexMeterB;
        phase = PhaseCode.B;
    } else {
        meter = triplexMeterC;
        phase = PhaseCode.C;
    }
    // distributes houses to channels, 20 per channel (1-n)
    int channelId = ((i - 1) / 20) + 1;
    generateHouse(sim, i, meter, triplineConf, auction,
        phase, controllerNIPrefix, channels.get(channelId));
}

```

## Reduce Complexity

## Reduce Error

## Programmability

```

interface "ZmqNetworkInterface",
"broker": "localhost",
"simulator_type": "power_grid",
"synchronization_algorithm":
"GracePeriodSyncAlso",
"sync_params": {
"number_of_power_grid_sims": 2
},
"simulator_time_metric": "seconds",
"packet_lost_period": 2300000000

```

```

classDefName: newPower_Backbone_N
...

```

```

...

```

```

...

```



```

...

```

```

use_override ON;
override override;
market Market1;
schedule_skew 2019;
bid_mode PROXY;
proxy_delay 10;
object controller network_interface {
    name F1_C_NI1;
    destination Market1NI;
};
control_mode RAMP;

```

```

Addresses.SetBase("10.1.2.0", "255.255.255
Addresses.Assign(p2pDevices_backbone_0);
push_back("F1_C_NI1");
push_back("F1_C_NI2");
push_back("Market1NI");
marketToControllerMap["Market1NI"] = "F1_C_NI";

```

## Simulators

# Questions

Arion Source Code - <https://github.com/pnnl/arion>

For more information contact:

Thomas W. Edgar

[thomas.edgar@pnnl.gov](mailto:thomas.edgar@pnnl.gov)