# **The Transactive Energy Challenge Abstract Component Model**

## **Executive Summary**

A transactive energy (TE) abstract component model (“TEM”) was the product of a "tiger team" effort engaged by the National Institute of Standards and Technology (NIST), Pacific Northwest National Lab (PNNL), Vanderbilt University, and Carnegie Mellon University during the summer of 2016 in support of the NIST Transactive Energy Challenge [1]. The purpose of this activity was to distill, from the collective experience of the participants, an abstract model of a transactive energy system consisting of an energy grid, loads, generators, controllers, and transactive agents.

The tiger team created TEM to provide a basis for common discussions, like the IEEE standard feeder models that the industry has used for more than a decade to test changes to simulations. The feeder models have allowed side-by-side comparisons, speeding up development of better simulation tools, and understanding of modeling and simulation in many university power engineering programs.

It is intended to assemble software and/or hardware implementations that realize the interfaces of this model. NIST will use its Universal CPS Environment for Federation (UCEF) built by NIST and Vanderbilt. Once realized it can be used for comparative studies of various transactive energy concepts. Alternatively, a platform developed by PNNL has been made available [2].

This report summarizes the detailed design of this component model, a canonical experiment into which model component realizations can be mounted, and an analysis of one typical testbed implementation by PNNL using GridLAB-D and how its contents can be mapped to the model.

Appendix A. References

1. 110th Congress Public Law 140 (2007), *Energy Independence and Security Act Of 2007*, retrieved from [https://www.gpo.gov/fdsys/pkg/PLAW-110publ140/html/PLAW-110publ140.htm](https://www.gpo.gov/fdsys/pkg/plaw-110publ140/html/plaw-110publ140.htm)
2. NIST Special Publication 1108r3, (2014), *NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0*, [http://dx.doi.org/10.6028/NIST.SP.1108r3](http://dx.doi.org/10.6028/nist.sp.1108r3)
3. GridWise Architecture Council (2015), *GridWise Transactive Energy Framework Version 1.0* , available at: <http://www.gridwiseac.org/pdfs/te_framework_report_pnnl-22946.pdf>
4. National Institute of Standards and Technology (NIST) Transactive Energy Challenge collaboration site. Retrieved from <https://pages.nist.gov/TEChallenge/>
5. Pacific Northwest National Labs (PNNL) Transactive Energy Simulation Platform (TESP). Retrieved from <http://tesp.readthedocs.io/en/latest/>
6. Universal CPS Environment for Federation (UCEF) project collaboration site. Retrieved from <https://pages.nist.gov/ucef/>
7. Holmberg, D., Hardin, D, Cunningham, R, Melton, R, Widergren, S, (2016) SGIP Technical Paper, *Transactive Energy Application Landscape Scenarios*, available at: <https://sepapower.org/resource/transactive-energy-application-landscape-scenarios/>
8. NIST (2018) TE Challenge Component Model: retrieved from <https://github.com/usnistgov/TEChallengeComponentModel>
9. PNNL-SA-113294 (2015), *Transactive Valuation Methodology Insights*, available at: <http://www.gridwiseac.org/pdfs/workshop_20150929/pnnl_sa_113294.pdf>.
10. NREL Typical Meteorological Year TMY(3), [http://rredc. nrel.gov/solar/old\_data/nsrdb/1991-2005/tmy3/](http://rredc.nrel.gov/solar/old_data/nsrdb/1991-2005/tmy3/)
11. ASHRAE (2016). Facility Smart Grid Information Model (FSGIM), ANSI/ASHRAE Standard 201-2016

## **Overview**

The evolving smart grid, with increased use of renewable energy generation and distributed energy management technologies, offers the potential for significant efficiency improvements through market-based transactive exchanges between energy producers and energy consumers. To understand this potential and support technology developers and policy makers, the smart grid community will require simulation tools and platforms that can be used to explore the benefits and impacts of alternative ways to create and operate energy systems. The National Institute of Standards and Technology (NIST) is charged by the 2007 Energy Independence and Security Act (EISA) [1] with facilitation of interoperability standards to enable successful implementation of the evolving cyber-physical national electric grid system known as the smart grid. As part of this effort, NIST has produced a framework [2] to describe its understanding of the state of the smart grid standards. The use of economic signals to mitigate demands was identified as key tool in grid evolution.

In this regard, transactive energy (TE) has been identified as an important area of research for the development of a modern electric grid with important technical challenges as well as needed interoperable approaches. The U.S. Department of Energy (DOE) Gridwise Architecture Council has published a Transactive Energy Framework [3] that defines TE broadly as, “a system of economic and control mechanisms that allows the dynamic balance of supply and demand across the entire electrical infrastructure using value as a key operational parameter.” To achieve this goal the Transactive Energy Modeling and Simulation Challenge for the Smart Grid (TE Challenge) [4] brings researchers and companies with simulation tools together with other grid stakeholders to demonstrate modeling and simulation platforms while applying TE approaches to real grid problems.

NIST is working in coordination with the DOE to explore the potential of TE to improve the safety, efficiency, reliability, resiliency, and adaptability of the grid. The NIST Transactive Energy Challenge Phases I & II aim to engage organizations with interests in TE to develop simulation-platform-agnostic common understandings and interoperable TE modeling approaches that will allow the broad community of electric grid and systems modelers to incorporate transactive elements into their own analyses and designs. The TE Challenge is designed to facilitate the application of common TE principles that can be explored with integrity across diverse modeling simulation toolsets.

TE is a complex system-of-systems problem and the detailed simulations require a diverse set of simulation tools. One of the key challenges for advancing TE is that of developing co-simulation platforms that can integrate multiple simulation tools to carry out a TE simulation. And a specific foundational element of that challenge is coming to some agreement at the conceptual level of the key components and interfaces for transactive energy simulations.

The transactive energy abstract component model (“TEM”) is this foundational model that aids understanding and communications of TE co-simulation. The model was the product of a tiger team effort engaged by the National Institute of Standards and Technology (NIST), Pacific Northwest National Lab (PNNL), Vanderbilt University, and Carnegie Mellon University during the summer of 2016 in support of the NIST Transactive Energy Challenge [1]. The purpose of this activity was to distill from the collective experience of the participants an abstract model of a transactive energy system consisting of an energy grid, loads, generators, controllers, and transactive agents.

The tiger team created this Model to provide a basis for common discussions, like the IEEE standard feeder models that the industry has used for more than a decade to test changes to simulations. The feeder models have allowed apples-to-apples comparisons, speeding up development of better simulation tools, and understanding of modeling and simulation in many university power engineering programs. The TE abstract component model should be able to provide similar support for TE in the industry. However, while the IEEE feeder models provide standard sets of physical components attached to a physical grid, the TE Model goes beyond this to describe abstract components with defined interfaces to allow connecting these components in simulation. A notional diagram of the Model is given in Figure 1.

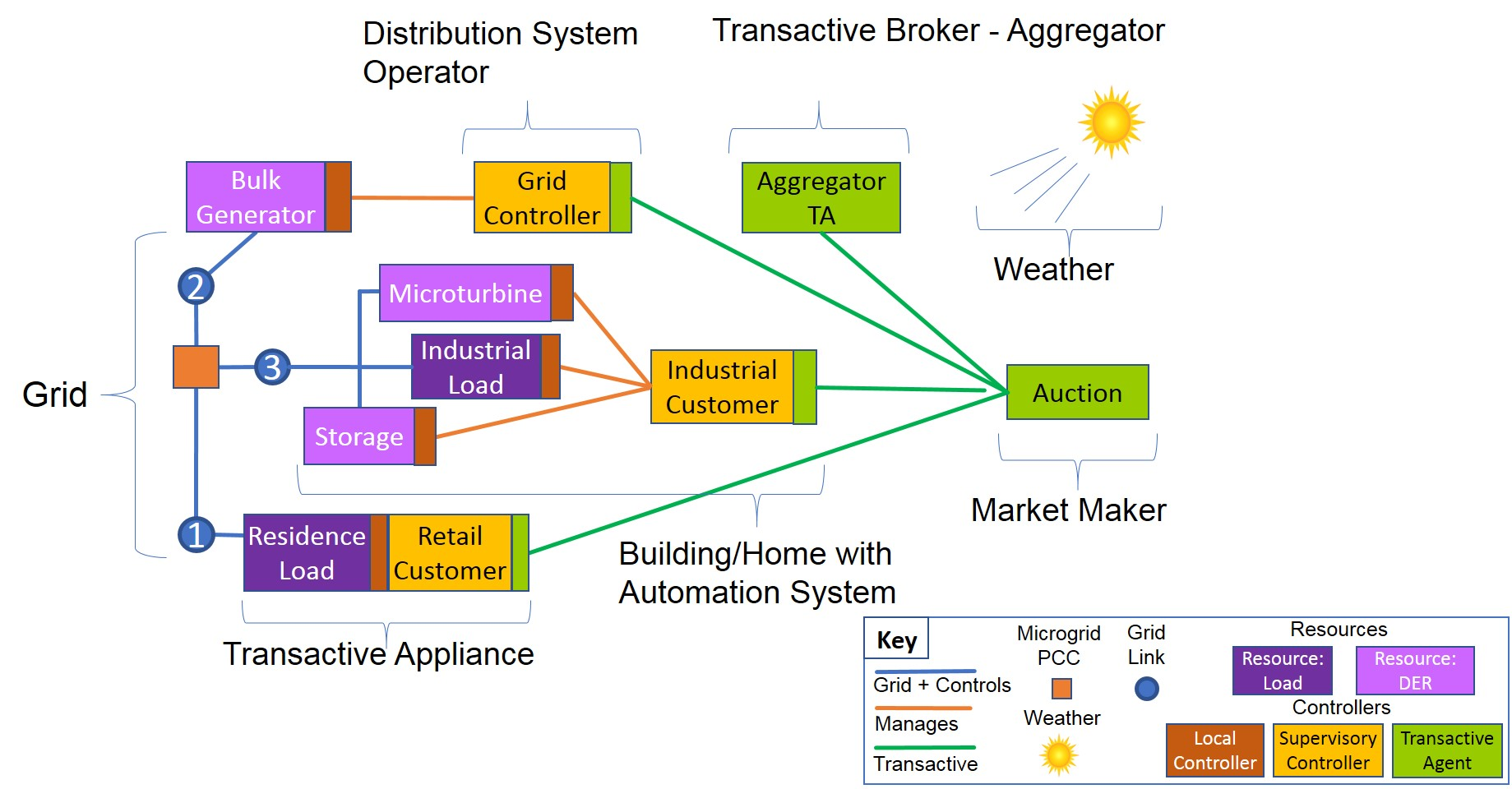


Figure 1. Notional Topology Illustrating TEM Building Blocks

Figure 1 illustrates how common devices and subsystems for energy can be composed of assemblies of relatively few re-usable building blocks. This reference diagram shows some typical assemblies. For example, a transactive appliance such as a hot water heater might consist of a load (the heater and tank), a local controller (i.e. a thermostat), a supervisory controller (that knows about the customer preferences such as schedule), and a transactive agent (that can bid for cost effective energy availability).

While Figure 1 shows a system that is much simpler than a real-world situation, it is complete from a testing point of view. There is at least one actor of each major type and one component of each major type. The Model can be used to evaluate any of the Smart Grid Interoperability Panel (SGIP) TE Landscape Scenarios [4], and any other use cases that support TE. The diagram illustrates the variations in realizations of components of a transactive energy system. This document goes on to describe a set of Model components that can be assembled and otherwise extended to simulate transactive systems of arbitrary design.

The goal of the Model is to enable understanding, discussion, evaluation and validation of any TE approach. Additionally, the Model can be used to study grid operations and controls as part of the TE approach. In a teaching environment, it can be used to test the student’s understanding and to frame the project that the student is working on within the TE context.

To understand the Model, consider that there are building blocks of resources, controllers, transactive agents, and the grid itself. These can be composed into various devices and systems. For example, a transactive appliance (Fig. 1) contains a load, a local controller, a supervisory controller, and a transactive agent. This composite device can participate in a transactive energy negotiation and regulate its operation accordingly to price signals.

TE is a complex system-of-systems problem and the detailed simulations require a diverse set of simulation tools. The Model allows connection of diverse simulation tools; a developer of simulation tools (e.g., for the building management system or a storage control system) may use the Model to enable connection of its tool into a complete TE test environment—that is, the grid modeling simulation software in addition to other components. This will enable developers to bring specific tools, models, and algorithms into standard baseline TE systems to test performance of these specific components against a common baseline TE environment.

The tiger team created a common platform with well-defined interfaces and semantics that stakeholders can understand and use to evaluate their own situations in their own context. The TE model makes it possible for stakeholders that do not understand the underlying grid model, or other specific technical aspects, to quickly understand TE and evaluate items that are important to them without having to have an expert help them set up and evaluate their specific aspect of TE.

Control simulations and algorithms can be layered on the TE model for testing. Control interactions that may be needed to balance the electrical network can be layered on to the TE model to investigate reactions and needed latency or lead times. Different pricing and economic approaches can be discussed and tested within the TE model framework. With enough customer behavior information, the economic models can be simulated so that one might understand how customers would react and the trigger prices that might be needed to achieve desired behavior. In addition, the Model allows the creating of tests of end-to-end security for each transaction both for control and communications.

In short, the TE Model provides a simplified environment to talk about, design and test almost any aspect of a TE approach. Note that this abstract component model does not specify an implementation. However, it provides the skeleton within which any given implementation can be discussed and compared on similar terms.

The overall TE model allows stakeholders to not only test items, but to design specific algorithms and tools that are proprietary. That means that competitors can use the same model to each test their specific competitive advantage and keep it secret from the competition, but at the same time be able to discuss in general terms what they are attempting to achieve with other stakeholders in a context that everyone understands.

In the long run, data sets, behavior models, common starting tools, message sets, communications modeling tools and other supporting tools will be developed on the TE model so that any stakeholder has a starting toolbox from which to tinker. Then, stakeholders can create their own improved tools for the specific areas they are interested in, while running the common tools for other aspects. This shortens development times for stakeholders and greatly lowers the barriers to entry for involvement.

The TE model can be extended by any stakeholder in any specific fashion to deal with simulation of some aspect. For instance, the TE model might be extended to all the appliances and other devices in a home for a stakeholder looking at home energy management systems.

An implementation of this Model in a simulation environment must faithfully implement Model interfaces. These interfaces can be extended as needed by the implementer. At the same time complexity can be hidden by combining components in the Model where the interfaces between the components are not important to the question(s) being evaluated.

The implementation orchestrates a set of components. These components, like the interfaces, can be extended or minimized depending on the experimental goals. Ideally the components can be simulated by the same experiment controller.

When the goal of an implementation is to enable comparisons using a common baseline, then the implementation also needs to support the defined set of grid nodes, resources, controllers, transactive agents, and market simulations required by the comparison baseline. In addition, the implementation must generate a core set of analytics that will enable evaluation of results from the implemented model against the baseline model and data.

The detailed abstract Model specification herein has been designed to be allow implementation on one or more simulation platforms. Many of the commonly used simulation platforms will support the implementation of this Model.

The balance of this Model document is organized into the following sections:

* Section 2: Model -- Describes the components that make up the abstract model
* Section 3: Scenario -- Describes a canonical scenario that can be run on the model components once realized on a suitable platform.
* Section 4: Beta Use Case -- A 30-house simple energy distribution model with double auction bidding is described
* Section 5: Composite and Extended Classes -- Illustrates how different concrete realizations can be built by using the standardized interfaces.

## **Model**

The TEM model consists of a set of abstract components for use in studying transactive energy. Each component represents a set of roles or interfaces that an actual device or computing platform might play in a transactive energy simulation. This section is divided into three parts:

* The core transactive component models;
* The interfaces that can be realized for interacting with the components; and
* The data types that flesh out the minimum detailed attributes that can be exchanged by the components

Throughout the balance of this model, most of the terminology is designed for use in software implementations based on the model. As such, names must have no spaces separating parts of compound names. Two typical methods are used in software for combining words to make compound names -- "camel case", and underscore delimited.

In TEM, names are done in camel case where each word is begun with a capital letter. By this means the phrase "by this means" becomes "ByThisMeans".

In GridLAB-D discussed in the Beta Use Case section below, underscore delimited compound naming is used. By this means becomes "by\_this\_means".

### **Transactive Energy Components**

The model components, shown in Figure 2, expose the key interfaces of the TEM. The model comprises the component roles, their interfaces, and data required (Note that in an actual implementation, several of these "roles" may be combined into a single instance exposing multiple interfaces. See examples in the "Composite Classes" section).

At the heart of the model is a simulation grid, Grid, which represents the electrical distribution system. There is a great deal of modeling and experience in describing grids. However, for this transactive energy abstraction, the grid represents the entirety of connected devices responsible for delivery and operation. It only exposes the links and nodes to which transactive energy resources are attached. Resource components, Resource, consist of loads and generators (including storage) that sink or source energy.

There are two types of controllers -- local and supervisory. The local controller, LocalController, is a component that understands the nature of a resource. A thermostat is a good example of a local controller for an HVAC system load. The component model concentrates the physical nature of resources in the resource definition and the logical part of the component in the local controller. This allows the "physical part" to interact with the physics of the grid simulation while the local controller can interact with the supervisory controller, SupervisoryController, in a higher abstraction of supervisory control.

There is a transactive agent, TransactiveAgent, that is typically tightly coupled to the supervisory controllers and is responsible for offering, bidding, and negotiating the price of energy.

Finally, a weather component, Weather, is responsible for providing the changing environment that drives energy production and consumption.

Two additional meta components -- the experiment manager, ExperimentManager, and, the analytics component, Analytics, represent the simulation test harness that orchestrates and analyzes the transactive energy scheme.

The balance of this section will describe the core components of the model.

#### **Transactive Energy Components diagram**

Figure 2 illustrates the classes or roles of components of the TEM. The diagram shows three groupings of model components:

* Core Components -- these represent the granularity of components that can be used in simulations. Additionally, they can be combined to represent less-granular components by aggregating their function and interfaces into composite components.
* Specializations -- these represent specializations of the core components for specific roles in transactive energy simulations.
* Experiment Orchestration and Analysis -- these represent the orchestrator for the simulation experiment (ExperimentManager), and, the core analytics component (Analytics) that evaluate the data produced during the simulation.

These components provide for a minimum of interoperability and the ability to define an experiment that can exercise the models based on these designs. Any realization of these models might extend their capabilities and information exchanged. The base interoperable characteristics herein provide for the consistency of simulation execution and minimal availability of data for the analytics.

The components in this model can be seen to have "lollypop" symbols -- a circle and a line -- sticking out of the left side of the component symbol. These represent the interfaces realized by the component. See Section 2.2 for the interface definitions.



Transactive Energy Components

#### **LocalController**

A simple base controller that contains logic to control a resource based on a dimensionless modulation setting received from a SupervisoryController. It does not have awareness of any other component of the system. Local Controllers include not only load and generator controllers, but also controllers of voltage regulators and protection devices and determine what state the device should be in and how it might progress to the next state. Example controllers include thermostats and loop controllers.

LocalController Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| actualDemand | float | This attribute defines the power being consumed by the device (as measured by the present subinterval demand) at the present time. |

|  |  |  |
| --- | --- | --- |
| demandLimits | PowerRatings | The operational demand characteristics and their associated end points for the load. |

|  |  |  |
| --- | --- | --- |
| downRamp | PowerRampSegmentType | This attribute defines the reduction in power over time when a load or generator being partially or fully de-energized has a complex load reduction profile. For each element of the Load.downRamp array, the downRamp[n].rate defines the amount of power decrease and the downRamp[n].duration defines the length of time in seconds upon which the decrease is in effect. |

|  |  |  |
| --- | --- | --- |
| locked | Boolean | This attribute defines whether the load is locked and therefore ineligible for curtailment; or unlocked and available for curtailment. Load locking behavior changes depending on the load's curtailmentStatus attribute value at the time the load was locked. |

|  |  |  |
| --- | --- | --- |
| status | LoadStatusType | This attribute defines the current status of the load. For non-curtailable loads, it provides the present communication status and reliability of the data. For curtailable loads, it also defines if the load is eligible for curtailment or why it is ineligible for curtailment. |

|  |  |  |
| --- | --- | --- |
| upRamp | PowerRampSegmentType | This attribute defines the increase in power over time when a load being partially or fully energized has a complex demand restoration profile. For each element of the Load.upRamp array, the upRamp[n].rate defines the amount of power increase and the upRamp[n].duration defines the length of time in seconds upon which the increase is in effect. |

#### **SupervisoryController**

Component that controls Resources indirectly by influencing LocalControllers; there does not need to be a one-to-one mapping of SupervisoryController to a resource (e.g., this may include a non-transactive aggregator or volt-var control system). The SupervisoryController acts by providing a dimensionless command (and other data in an extended class if so implemented) to a LocalController. It is the LocalController that knows the details of the Resource to be controlled. The SupervisoryController has awareness of all the Resources it is responsible for monitoring and controlling.

This explicitly does not contain transactive elements but does contain non- transactive optimizations and operator and consumer actions. The implementation of this component may be extremely broad, since there are many different control variables that may drive a supervisory control algorithm. For example, a building control system which has setpoints and schedules, and other customer preferences may act as the SupervisoryController for all Resources in a home, building or campus.

SupervisoryController Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| resources | Resource | This attribute defines the list of loads, generators, storage devices this SupervisoryController manages. |

#### **Resource**

Grid connected Resources can be loads, generators, or storage devices. They consume or generate energy. Resources are considered to be intelligent in that they can provide information about themselves and have a defined interface for local control over their operation. In practice, there may be many details of a resource that a modeler may expose. Those shown in this model are minimum interfaces required to perform standardized simulations for transactive energy. Actual models will inherit from these more general interfaces to include the specialized behaviors and information exchanges.

Resource Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| current | Current | Current consumed (positive) by Resource by phase. |

|  |  |  |
| --- | --- | --- |
| gridNodeId | char | Identifies grid node load is connected to. |

|  |  |  |
| --- | --- | --- |
| impedance | Impedance | Resource complex impedance by phase and across phases. |

|  |  |  |
| --- | --- | --- |
| phases | Phases | Electrical phases. |

|  |  |  |
| --- | --- | --- |
| power | Power | Power consumed (positive) by Resource by phase. |

|  |  |  |
| --- | --- | --- |
| status | boolean | Status of resource -- active (true) or inactive (false). |

|  |  |  |
| --- | --- | --- |
| voltage | Voltage | Voltage by phase. |

#### **Weather**

The Weather component provides environmental conditions during a TE Challenge experiment. The properties of the weather component are based on the National Solar Radiation Data Base from NREL know as Typical Model Year (TMY)3 [10].

Weather Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| aerosol\_optical\_depth | double | The broadband aerosol optical depth per unit of air mass due to extinction by the aerosol component of the atmosphere.  Unit: AOD [unitless]  Resolution: 0.001 |

|  |  |  |
| --- | --- | --- |
| aerosol\_optical\_depth\_source | char | AOD source. |

|  |  |  |
| --- | --- | --- |
| aerosol\_optical\_depth\_uncertainty | double | AOD uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| albedo | double | The ratio of reflected solar irradiance to global horizontal irradiance.  Unit: Alb [unitless]  Resolution: 0.01 |

|  |  |  |
| --- | --- | --- |
| albedo\_source | char | Alb source. |

|  |  |  |
| --- | --- | --- |
| albedo\_uncertainty | double | Alb uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| latitude | double | Site latitude.  Unit: Decimal degree |

|  |  |  |
| --- | --- | --- |
| ceiling\_height | double | Height of the cloud base above local terrain (77777 = unlimited).  Unit: CeilHgt (m)  Resolution: 1m |

|  |  |  |
| --- | --- | --- |
| ceiling\_height\_source | char | CeilHgt source. |

|  |  |  |
| --- | --- | --- |
| ceiling\_height\_uncertainty | double | CeilHgt uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| date | char | Date of data record.  Unit: mm/dd/yyyy |

|  |  |  |
| --- | --- | --- |
| dew\_point\_temperature | double | Dew-point temperature at the time indicated.  Unit: Dew-point (Degree C)  Resolution: 0.1degree |

|  |  |  |
| --- | --- | --- |
| dew\_point\_temperature\_source | char | Dew-point source. |

|  |  |  |
| --- | --- | --- |
| dew\_point\_temperature\_uncertainty | double | Dew-point uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_illuminance | double | Average amount of illuminance received from the sky (excluding the solar disk) on a horizontal surface during the 60-minute period ending at the timestamp.  Unit: DH illum (lux)  Resolution: 100 Ix |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_illuminance\_source | char | DH illum source. |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_illuminance\_uncertainty | double | DH illum uncertainty (%). |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_irradiance | double | Amount of solar radiation (modeled) received in a collimated beam on a surface normal to the sun during the 60-minute period ending at the timestamp.  Unit: DHI (W/m^2)  Resolution: 1Wh/m^2 |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_irradiance\_source | char | DHI source. |

|  |  |  |
| --- | --- | --- |
| diffuse\_horizontal\_irradiance\_uncertainty | double | DHI uncertainty (%). |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_illuminance | double | Average amount of direct normal illuminance received within a 5.7° field of view centered on the sun during 60-minute period ending at the timestamp.  Unit: DN illum (lux)  Resolution: 100 IX |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_illuminance\_source | char | DN illum source. |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_illuminance\_uncertainty | double | Uncertainty based on random and bias error estimates.  Unit: DN illum uncertainty (%)  Resolution: 1% |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_irradiance | double | Amount of solar radiation (modeled) received in a collimated beam on a surface normal to the sun during the 60-minute period ending at the timestamp.  Unit: DNI (W/m^2)  Resolution: 1 Wh/m^2 |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_irradiance\_source | char | DNI source. |

|  |  |  |
| --- | --- | --- |
| direct\_normal\_irradiance\_uncertainty | double | DNI uncertainty (%). |

|  |  |  |
| --- | --- | --- |
| dry\_bulb\_temperature | double | Dry-bulb temperature at the time indicated.  Unit: Dry-bulb (Degree C)  Resolution: 0.1 degree |

|  |  |  |
| --- | --- | --- |
| dry\_bulb\_temperature\_source | char | Dry-bulb source. |

|  |  |  |
| --- | --- | --- |
| dry\_bulb\_temperature\_uncertainty | double | Dry-bulb uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| elevation | double | Site elevation.  Unit: Meter |

|  |  |  |
| --- | --- | --- |
| extra\_terrestrial\_radiation | double | Amount of solar radiation received on a horizontal surface at the top of the atmosphere during the 60-minute period ending at the timestamp.  Unit: W/m^2  Resolution: 1Wh/m^2 |

|  |  |  |
| --- | --- | --- |
| extra\_terrestrial\_radiation\_normal | double | Amount of solar radiation received on a surface normal to the sun at the top of the atmosphere during the 60-minute period ending at the timestamp.  Unit: W/m^2  Resolution: 1 Wh/m^2 |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_illuminance | double | Average total amount of direct and diffuse illuminance received on a horizontal surface during the 60-minute period ending at the timestamp.  Unit: GH illum (lux)  Resolution: 100 Ix |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_illuminance\_source | char | GH illum source. |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_illuminance\_uncertainty | double | Global illum uncertainty (%). |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_irradiance | double | Total amount of direct and diffuse solar radiation received on a horizontal surface during the 60-minute period ending at the timestamp.  Unit: W/m^2  Resolution: 1Wh/m^2 |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_irradiance\_source | char | GHI source. |

|  |  |  |
| --- | --- | --- |
| global\_horizontal\_irradiance\_uncertainty | double | GHI uncertainty (%). |

|  |  |  |
| --- | --- | --- |
| horizontal\_visibility | double | Distance to discernable remote objects at the time indicated (7777 = unlimited).  Unit: Hvis (m)  Resolution: 1m |

|  |  |  |
| --- | --- | --- |
| horizontal\_visibility\_source | char | Hvis source. |

|  |  |  |
| --- | --- | --- |
| horizontal\_visibility\_uncertainty | double | Hvis uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| liquid\_precipitation\_depth | double | The amount of liquid precipitation observed at the indicated time for the period indicated in the liquid precipitation quantity field.  Unit: Lprecip depth (mm).  Resolution: 1mm |

|  |  |  |
| --- | --- | --- |
| liquid\_precipitation\_depth\_source | char | Lprecip source. |

|  |  |  |
| --- | --- | --- |
| liquid\_precipitation\_depth\_uncertainty | double | Lprecip uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| liquid\_precipitation\_quantity | double | The period of accumulation for the liquid precipitation depth field.  Unit: Lprecip quantity (hr)  Resolution: 1hr |

|  |  |  |
| --- | --- | --- |
| longitude | double | station longitude.  Unit: Decimal Degree |

|  |  |  |
| --- | --- | --- |
| opaque\_sky\_cover | double | Amount of sky dome covered by clouds or obscuring phenomena that prevent observing the sky or higher cloud layers at the time indicated.  Unit: OpqCld (tenths)  Resolution: 1 tenth |

|  |  |  |
| --- | --- | --- |
| opaque\_sky\_cover\_source | char | OpqCld source. |

|  |  |  |
| --- | --- | --- |
| opaque\_sky\_cover\_uncertainty | double | OpqCld uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| precipitable\_water | double | The total precipitable water contained in a column of unit cross section extending from the earth's surface to the top of the atmosphere.  Unit: Pwat (cm)  Resolution: 0.1cm |

|  |  |  |
| --- | --- | --- |
| precipitable\_water\_source | char | Pwat source. |

|  |  |  |
| --- | --- | --- |
| precipitable\_water\_uncertainty | double | Pwat uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| present\_weather | double | PresWth (METAR code). |

|  |  |  |
| --- | --- | --- |
| present\_weather\_source | char | PresWth source. |

|  |  |  |
| --- | --- | --- |
| present\_weather\_uncertainty | double | PresWth uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| pressure | double | Station pressure at the time indicated.  Unit: Pressure (mbar)  Resolution: 1mbar |

|  |  |  |
| --- | --- | --- |
| pressure\_source | char | Pressure source. |

|  |  |  |
| --- | --- | --- |
| pressure\_uncertainty | double | Pressure uncert (code). |

|  |  |  |
| --- | --- | --- |
| relative\_humidity | double | Relative humidity at the time indicated.  Unit: RHum (%)  Resolution: 1% |

|  |  |  |
| --- | --- | --- |
| relative\_humidity\_source | char | RHum source. |

|  |  |  |
| --- | --- | --- |
| relative\_humidity\_uncertainty | double | RHum uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| station\_id\_code | int | station identifier code. |

|  |  |  |
| --- | --- | --- |
| station\_name | char | Station name. |

|  |  |  |
| --- | --- | --- |
| station\_state | char | Station state. |

|  |  |  |
| --- | --- | --- |
| time | char | hh:mm:ss local time. |

|  |  |  |
| --- | --- | --- |
| time\_zone | double | Station time zone. Hours from Greenwich, negative west. |

|  |  |  |
| --- | --- | --- |
| total\_sky\_cover | double | Amount of sky dome covered by clouds or obscuring phenomena at the time indicated.  Unit: TotCld (tenths)  Resolution: 1 tenth |

|  |  |  |
| --- | --- | --- |
| total\_sky\_cover\_source | char | TotCld source. |

|  |  |  |
| --- | --- | --- |
| total\_sky\_cover\_uncertainty | double | TotCld uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| wind\_direction | double | Wind direction at the time indicated.  Unit: Wdir (degrees)  Resolution: 10 degree |

|  |  |  |
| --- | --- | --- |
| wind\_direction\_uncertainty | double | Wdir uncert (code). |

|  |  |  |
| --- | --- | --- |
| wind\_direction\_source | char | Wdir source. |

|  |  |  |
| --- | --- | --- |
| wind\_speed | double | Wind speed at the time indicated.  Unit: Wspd (m/s)  Resolution: 0.1 m/s |

|  |  |  |
| --- | --- | --- |
| wind\_speed\_source | char | Wspd source. |

|  |  |  |
| --- | --- | --- |
| wind\_speed\_uncertainty | double | Wspd uncertainty (code). |

|  |  |  |
| --- | --- | --- |
| zenith\_luminance | double | Average amount of luminance at the sky's zenith during the 60-minute period ending at the timestamp.  Unit: Zenith lum (cd/m^2)  Resolution: 10 cd/m^2 |

|  |  |  |
| --- | --- | --- |
| zenith\_luminance\_source | char | Zenith lum source. |

|  |  |  |
| --- | --- | --- |
| zenith\_luminance\_uncertainty | double | Uncertainty based on random and bias error estimates.  Unit: Zenith lum uncertainty (%)  1% |

#### **Grid**

A simulation of a power grid or grid segment. A Grid consists of a set of Link structures that represent a network of interconnected nodes that comprise and energy system. Resources such as loads and generators are "attached" to the nodes of the Grid.

Grid Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| Nodes | Link | List of nodes in Grid. See definition of Link in the DataTypes section. |

#### **TransactiveAgent**

Describes transactions in terms of location for delivery, with agreed on delivery time window, product (e.g., real power or transport), and expression of value (e.g., price) with logic for estimating that value (e.g., forecasts) and quantity (including limits), and rules for how “bids” are formed and how often they are presented. This would include all “market” functions including device-level bidding (replacing a traditional controller/thermostat), large-scale optimization (e.g., it could be used to describe an ISO or double-auction), etc.

A given TransactiveAgent component is the realization of a market pricing mechanism and obtains demands and forecasts from SupervisoryControllers and negotiates pricing with other TransactiveAgents. Note that some TransactiveAgents may be market makers and others may be responsive to SupervisoryControllers and interact with the market makers. Other TransactiveAgents may be modeled for peer to peer price negotiations without need for a market maker instance.

TransactiveAgent Detail

#### **Load**

A specialization of a resource that represents a customer premise based load.

Load Detail

#### **Generator**

A specialization of a resource that represents a customer premise-based grid connected generation source.

Generator Detail

#### **GridController**

A specialization of the SupervisoryController that provides for supervisory control of the grid segment and represents the distribution system operator (DSO) or similar entity that can represent the grid management in a transactive energy scheme.

GridController Detail

#### **Auction**

A specialization of the TransactiveAgent that is essentially the market maker or broker. Some transactive energy schemes are purely peer to peer. They do not need an auction component. Others require a central component where participants can contribute their offerings and bids and that conducts the algorithm by which pricing is determined from the collective participants.

Auction Detail

#### **ExperimentManager**

The ExperimentManager runs the experiment. It is typically responsible for providing initialization data for the individual components and managing the time progression of an experiment.

ExperimentManager Detail

#### **Analytics**

Analyzes data and produces metrics of the scenario. Each instance of Analytics is generated for a step in the time sequence of the experiment based on the timing of messaging for each phase of the message loop.

Analytics Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| aggregatedLoadsByHousehold | Energy | Aggregated load by household. |

|  |  |  |
| --- | --- | --- |
| generationProfile | Energy | Generation by generator. |

|  |  |  |
| --- | --- | --- |
| gridPower | Power | Power provided by the Grid. |

|  |  |  |
| --- | --- | --- |
| loadProfile | Energy | Energy consumed by each load. |

|  |  |  |
| --- | --- | --- |
| priceNegotiations | Tender | Sequence of all tenders. |

|  |  |  |
| --- | --- | --- |
| realizeMarketPricing | Quote | Realized Market price quotes. |

|  |  |  |
| --- | --- | --- |
| voltage | Voltage | Voltage at every link point in pairs in link order and fromVoltage prior to toVoltage. |

#### **BaseModelComponent**

General Transactive Energy Model Component. This abstract component provides for the initialization of simulation model components.

BaseModelComponent Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| description | char | General Transactive Energy Model Component provides for the initialization of simulation model components. |

|  |  |  |
| --- | --- | --- |
| name | char | Name of the component |

### **Interfaces**

The model is designed with component models that have defined interfaces. The use of interfaces, as opposed to component class methods, makes aggregation of function possible without changing the designs. For example, a composite device such as a transactive appliance, if implemented, would realize all the interfaces shown for the Resource, LocalController, SupervisoryController, and TransactiveAgent. By this means, the participating component realizations in the experiment can be agnostic to the underlying class model implementations.

These interfaces, shown in Figure 3, represent those messages that are received (subscribed) by the realizing software. It is assumed that to invoke these interfaces, the source can publish the data.

In implementing this model, pub-sub or request-response can produce equivalent results and the arrows and data flows should be interpreted appropriately to the underlying message mechanism.

Interface naming in software uses the convention of prefixing a camel case name with a lower case "i". So that the TA interface name becomes "iTA".

#### **Interfaces diagram**

Figure 3 illustrates the interfaces defined for the TEM. Shown are the core components of the model and those interfaces that they realize.

Each interface exchanges a named type data structure. The details of this data can be found in the Data Types section which follows.



Interfaces

#### **iTA**

The iTA interface is used by TransactiveAgent components to negotiate price from provided information from other participating TAs. Note that there is often an intimate relationship between software in SupervisoryControllers and TAs that allow the TAs to have the information needed to construct tenders and transactions.

iTA Detail

#### **iResourcePhysicalStatus**

This interface allows the Resource to report on its physical state. For example, the power consumed or generated.

iResourcePhysicalStatus Detail

#### **iGrid**

This interface allows management of the Grid and its ability to resolve its electrical state with connected load and generator resources.

iGrid Detail

#### **iWeatherData**

This interface allows subscribing components to accept a weather data feed.

iWeatherData Detail

#### **iLocalControl**

This interface provides a supervisory control signal to a LocalController.

iLocalControl Detail

#### **iResourceControl**

This interface allows a LocalController to control a Resource. For example, a local controller might receive measurements such as temperature from a hot water heater load. The control of relays controlling energy usage by the load are commanded by the LocalController to the Resource. Note that there are a very large degree of potential data that must be exchanged over this interface for any particular kind of resource and this will not be standardized in this specification to any degree. Note: in many cases, the LocalController and Resource will be combined into a single device -- e.g. a hot water heater (load resource) comes assembled including its thermostat (local controller).

iResourceControl Detail

#### **iResourcePhysical**

This interface allows the Grid simulation to provide physical state information to the Resource based on the physics computations to resolve grid state at any moment in the simulation.

iResourcePhysical Detail

#### **iGeneral**

General Interface for all components including Initialization.

iGeneral Detail

### **DataTypes**

Data Types used in message exchanges. This section details the attributes used in interface exchange and class definitions.

#### **DataTypes diagram**

This diagram presents data types in three categories:

* Component Model Data Structures -- defines the main data structures that represent grid state;
* Interface Parameter Classes -- defines the content of interface messages; and
* Data descriptions imported from ASHRAE 201 -- Facility Smart Grid Information Model (FSGIM) [11]. These data definitions were modified from the standard to fit the data primitives of this modeling effort. They can be losslessly converted from FSGIM data types.



DataTypes

#### **DataEnumerations diagram**

This diagram presents enumerations and primitive data types that are part of the model.



DataEnumerations

#### **Energy**

A complex vector of energy by phase.

Energy Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| energyByPhase | ComplexNumber | Energy by phase. |

#### **GridNode**

This is a node in the grid.

GridNode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| current | Current | Current flowing into the node. |

|  |  |  |
| --- | --- | --- |
| id | GridNodeId | This is id of grid node. |

|  |  |  |
| --- | --- | --- |
| power | Power | Power into the node. |

|  |  |  |
| --- | --- | --- |
| voltage | Voltage | Voltages at node. |

#### **AttachNodeDescription**

Parameters to attach a node to the Grid.

AttachNodeDescription Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| gridNodeID | char | Node identifier to attach to in the Grid model. |

|  |  |  |
| --- | --- | --- |
| impedance | Impedance | Impedance matrix for phase connections. |

|  |  |  |
| --- | --- | --- |
| phases | Phases | Phases for attachment. |

#### **ComplexNumber**

A complex number.

ComplexNumber Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| imaginary | float | The imaginary part. |

|  |  |  |
| --- | --- | --- |
| real | float | The real part. |

#### **Current**

A complex set of currents by phase, A.

Current Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| currentByPhase | ComplexNumber | Current by phase. |

#### **GridControl**

GridControl service data structure. Not defined at this time.

GridControl Detail

#### **GridVoltageState**

Details of voltage by phase at a node in the grid.

GridVoltageState Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| phases | Phases | Phase. |

|  |  |  |
| --- | --- | --- |
| voltage | Voltage | Voltage. |

#### **Impedance**

A set of complex impedances by phase and across phases.

Impedance Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| impedanceByPhase | ComplexNumber | Impedance by phase. |

#### **Link**

Describes the physical components of the system and their internal state properties, such as power (or current) flow, current tap position, etc. It also includes topological information, such as “to” and “from”, which makes the nodal information implicit. Note that "to" and "from" simply identify two ends of the link and do not make a statement about the direction of flow of power/energy.

Link Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| fromGridNode | GridNodeId | One connecting end of the link object. This will be the reference to a node-based object elsewhere in the powerflow model. |

|  |  |  |
| --- | --- | --- |
| id | char | Link segment id. |

|  |  |  |
| --- | --- | --- |
| impedance | Impedance | Matrix of impedances for each phase. |

|  |  |  |
| --- | --- | --- |
| length | float | Length of link segment in meters. |

|  |  |  |
| --- | --- | --- |
| name | char | Link name. |

|  |  |  |
| --- | --- | --- |
| phases | Phases | Sequence of phases from A, B, C, N, L1, L2. e.g. three phase 4 wire -- ABCN Note phase order indicates the index in to the vector array of impedances or power or voltage. Voltage relative to N. |

|  |  |  |
| --- | --- | --- |
| status | boolean | True if connectivity established false if connectivity denied. |

|  |  |  |
| --- | --- | --- |
| toGridNode | GridNodeId | Identifies the second node of the link. |

#### **PiecewiseLinearSegment**

The PiecewiseLinearSegment class defines the attributes needed to specify a single straight line segment for a piecewise linear curve. Each straight line segment is specified by two X-axis coordinates, percentOfFullRatedOutputBegin and percentOfFullRatedOutputEnd; and by two Y-axis coordinates, percentOfFullRatedInputPowerDrawnBegin and percentOfFullRatedInputPowerDrawnEnd. The entire piecewise linear curve is defined by the runningProfile attribute; where the 'percent of full rated input power' is a function of the 'percent of full rated output'. That is, as the output varies between 0 and 100 percent; the function maps to the percentage of input power (0..100) required to achieve the specified output.

PiecewiseLinearSegment Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| desiredFractionOfFullRatedOutputBegin | float | This attribute defines the starting x-coordinate of the straight line segment. |

|  |  |  |
| --- | --- | --- |
| desiredFractionOfFullRatedOutputEnd | float | This attribute defines the ending x-coordinate of the straight line segment. |

|  |  |  |
| --- | --- | --- |
| requiredFractionOfFullRatedInputPowerDrawnBegin | float | This attribute defines the starting y-coordinate of the straight line segment. |

|  |  |  |
| --- | --- | --- |
| requiredFractionOfFullRatedInputPowerDrawnEnd | float | This attribute defines the ending y-coordinate of the straight line segment. |

#### **Power**

A complex vector of power by phase.

Power Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| powerByPhase | ComplexNumber |  |

#### **PowerCurve**

This class describes the characteristics of a mathematical function used to estimate the power consuming characteristics of a load or the power generating characteristics of a generator.

PowerCurve Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| maximumReactivePower | float | This attribute defines the maximum reactive power consumed (or supplied) by the device in units specified in PowerReactiveType. |

|  |  |  |
| --- | --- | --- |
| maximumRealPower | float | This attribute defines the maximum real power consumed (or supplied) by the device in units specified in PowerRealType. |

|  |  |  |
| --- | --- | --- |
| reactivelPowerCurve | PiecewiseLinearSegment | This attribute defines the reactive component of a single piecewise linear curve mapping the percentage of power consumed by the device as a function of the present level of operation of the device. |

|  |  |  |
| --- | --- | --- |
| realPowerCurve | PiecewiseLinearSegment | This attribute defines the real component of a single piecewise linear curve mapping the percentage of power consumed by the device as a function of the present level of operation of the device. |

#### **PowerRampSegmentType**

The PowerRampSegmentType data structure is used to define a single array element of the recoveryRamp and stagingRamp array of the Load class. Each array element defines the beginning demand for the line segment and the rate of rise or drop. These attributes combined with the duration completely forms a line segment defining a portion of the ramp.

PowerRampSegmentType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| beginRamp | float | The attribute defines the quantity of power at the start of the ramp segment. If this attribute is not defined in the segment, the start of the ramp is assumed to be the end of the ramp of the previous segment. |

|  |  |  |
| --- | --- | --- |
| duration | int | The attribute defines the time horizon in seconds upon which the associated rise or drop is valid. |

|  |  |  |
| --- | --- | --- |
| rampToCompletion | boolean | The attribute defines whether the ramping up or down of this load may be halted in midstream (false) or once started must complete through all segments of the ramp (true). As an example, a multistate fan may only use a portion of the ramp, as it sequences from low to medium to high speed levels (false); whereas, a production line, once started, may need to run through its complete set of ramp segments (true). If the attribute is not defined it is assumed to be false. |

|  |  |  |
| --- | --- | --- |
| rate | float | The attribute defines rate of rise (positive value) in demand or the rate of drop (negative value) in demand when a load either powers up or shuts down respectively. Its sister attribute, duration, defines the time frame upon which the rate is defined. |

#### **PowerRatings**

This class describes the power characteristics of a Load (or Generator) component. The attributes defined allow specifying the minimum and maximum expected power draw from the load (supply from a generator). It also allows a series of predefined operation power curves to be defined with one designated as presently being operational.

PowerRatings Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| activePowerCurve | int | This attribute defines the index into the zero based array of powerCurves indicating which powerCurve is presently active. |

|  |  |  |
| --- | --- | --- |
| adjustedFullDRPower | float | This attribute defines the minimum expected power draw of a load (or the maximum power supplied by a generator) during operation. This value differs from the rated power since it may take into account operational considerations such as environmental, equipment safety or regulatory conditions. |

|  |  |  |
| --- | --- | --- |
| adjustedNoDRPower | float | This attribute defines the maximum expected power draw of a load (or the minimum power supplied by a generator) during operation. This value differs from the rated power since it may take into account operational considerations such as environmental, equipment safety or regulatory conditions. |

|  |  |  |
| --- | --- | --- |
| powerCurves | PowerCurve | This attribute defines one or more piecewise linear curves mapping the percentage of power consumed by the device as a function of the present level of operation of the device. Many loads draw power (or generators supply power) based on the present loading characteristics of the device. For example, a motor driving a fan will draw more power as the fan blade pitch is increased. The axes of the curve are defined in percent to allow loads of any type to utilize the attribute.  When powerCurve is not present, the load or generator is assumed to be a two state device drawing no power when the device is off and adjustedNoDRPower when the device is on. When adjustedNoDRPower also is not present, the load or generator is assumed to be a two state device drawing no power when the device is off and maximumRealPower when the device is on. |

#### **PriceCurve**

A price curve. Depending on sign of the PriceCurveComponent.quantity it can be price of supply (+) or demand(-) correspondingly.

PriceCurve Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| priceCurveComponent | PriceCurveComponent | A component of a price curve. |

#### **PriceCurveComponent**

A component of a pricing curve.

PriceCurveComponent Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| price | float | Price of commodity. |

|  |  |  |
| --- | --- | --- |
| quantity | float | Quantity of commodity (signed number) can be supply (+) or demand (-). |

|  |  |  |
| --- | --- | --- |
| type | char | Type of commodity -- W, Var, V, Frequency, Wh, .... |

#### **Quote**

A quote exchanged by the TransactiveAgents.

Quote Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| quote | TenderComponent | Array of tender components. |

#### **ResourceControl**

Parameters used by local controller to control the resource. This class may be extended to provide additional information to the Resource in order to manage its state. No default information defined in this model.

ResourceControl Detail

#### **ResourcePhysicalState**

ResourcePhysicalState describes physical state parameters for the resource.

ResourcePhysicalState Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| phases | Phases | Phase. |

|  |  |  |
| --- | --- | --- |
| power | Power | Power. |

#### **ResourceStatus**

The status of the resource provided describing current load and other conditions.

ResourceStatus Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| current | Current | Current. |

|  |  |  |
| --- | --- | --- |
| phases | Phases | Phase |

|  |  |  |
| --- | --- | --- |
| power | Power | Power. |

|  |  |  |
| --- | --- | --- |
| status | boolean | Indicates if resource is active (true) or inactive (false). |

|  |  |  |
| --- | --- | --- |
| voltage | Voltage | Voltage. |

#### **SupervisoryControlSignal**

Supervisory control signal provided to LocalController.

SupervisoryControlSignal Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| modulationSignal | float | Modulation control signal 0..1.0 for off (0.0) to full load or supply (1.0). |

#### **Tender**

Tender data structure exchanged by TransactiveAgent during negotiation.

Tender Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| tenderComponent | TenderComponent | Array of time ordered tender components that provides a load or generation profile of price curves. |

#### **TenderComponent**

A component of a tender.

TenderComponent Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| priceCurve | PriceCurve | Price curve for this time reference. |

|  |  |  |
| --- | --- | --- |
| timeReference | int | Time reference for this tender component. |

#### **Transaction**

A transaction used to establish a price between TransactiveAgents.

Transaction Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| accept | boolean | Accept the last quote. |

#### **Voltage**

A complex vector of Voltage and a string enumeration of the phases

Voltage Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| voltageByPhase | ComplexNumber | Voltage by phase. |

#### **TimeReference**

A time reference, UTC.

TimeReference Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| t | int | Time reference. |

#### **GridNodeId**

An identifier representing a grid node.

GridNodeId Detail

#### **Phases**

The phases property has a variety of valid inputs. These are:

* A - Phase A of a three phase connection
* B - Phase B of a three phase connection
* C - Phase C of a three phase connection
* D - Delta connected phases - this implies ABC, but explicitly specifying them is recommended
* N - Neutral phase
* G - Ground phase
* S - Split phase - this represents residential level wires (2 "hot" and 1 "neutral" wire)

These different phases can be combined specified in combinations to represent phasing in power line segments, circuits and transformers. For example 3-Phase Electric Power with a neutral would be "ABCN". Two-phase residential power is typically "S12N".

Phases Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| A |  | A - Phase A of a three phase connection |

|  |  |  |
| --- | --- | --- |
| B |  | B - Phase B of a three phase connection |

|  |  |  |
| --- | --- | --- |
| C |  | C - Phase C of a three phase connection |

|  |  |  |
| --- | --- | --- |
| ABC |  | D - Delta connected phases - this implies ABC, but explicitly specifying them is recommended |

|  |  |  |
| --- | --- | --- |
| N |  | N - Neutral phase |

|  |  |  |
| --- | --- | --- |
| G |  | G - Ground phase |

|  |  |  |
| --- | --- | --- |
| S |  | S - Split phase - this represents residential level wires (2 "hot" and 1 neutral wire) |

|  |  |  |
| --- | --- | --- |
| AB |  | This implies AB. |

|  |  |  |
| --- | --- | --- |
| BC |  | This implies BC. |

|  |  |  |
| --- | --- | --- |
| AC |  | This implies AC. |

#### **StorageType**

An enumeration that defines the energy storage characteristics of an instance of the Generator Class.

StorageType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| none |  | This value indicates that this instance of the Generator Class models a device that does not produce energy from storage. |

|  |  |  |
| --- | --- | --- |
| electricalStorage |  | This value indicates that this instance of the Generator Class models a device that produces electricity from storage. |

|  |  |  |
| --- | --- | --- |
| thermalStorage |  | This value indicates that this instance of the Generator Class models a device that produces thermal energy from storage. |

#### **SupplyStatusType**

This enumeration indicates if the load is presently curtailed and if curtailed is in compliance with the curtailment request received.

SupplyStatusType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| supplyInactive |  | This generator is not presently operating |

|  |  |  |
| --- | --- | --- |
| supplyRequestPending |  | A request has been received and is pending. |

|  |  |  |
| --- | --- | --- |
| supplyCompliant |  | The generator operation is compliant with the last request. |

|  |  |  |
| --- | --- | --- |
| supplyNoncompliant |  | The generator is not compliant with the last request. |

#### **LoadStatusType**

This enumeration provides the present overall state of the load.

LoadStatusType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| NA |  | Not applicable |

|  |  |  |
| --- | --- | --- |
| eligible |  | The load is presently communicating properly and its data values are correct. In addition, for curtailable loads this load is presently eligible to be curtailed. |

|  |  |  |
| --- | --- | --- |
| loadLocked |  | The load is ineligible for curtailment since it has been locked. |

|  |  |  |
| --- | --- | --- |
| loadOverridden |  | An external process has set this override attribute prohibiting curtailment. |

|  |  |  |
| --- | --- | --- |
| lostCommunication |  | The load presently cannot be accessed. |

|  |  |  |
| --- | --- | --- |
| maxCurtailQueue |  | The load is in curtailment and presently being timed for the maximum curtailment time. |

|  |  |  |
| --- | --- | --- |
| maxCyclesThisPeriod |  | The load has been cycled the maximum number of times this period. |

|  |  |  |
| --- | --- | --- |
| minCurtailQueue |  | The load is in curtailment and presently being timed for the minimum curtailment time. |

|  |  |  |
| --- | --- | --- |
| pointNotConsumingEnergy |  | The load is ineligible for curtailment since the point associated with the load is already shut off and not consuming any energy. |

|  |  |  |
| --- | --- | --- |
| pointUnreliable |  | The load is ineligible for curtailment since the point associated with the load is unreliable. 'Unreliable' is an error condition when the present value of a point is questioned due to some hardware or software failure. When a point is unreliable, it still may present a value (e.g., Space Temp Present Value = 67 DegF) but carries along a second attribute that indicates this value is suspect. When a point is in the unreliable state, it shall not be curtailed. |

|  |  |  |
| --- | --- | --- |
| ramping |  | The load is ramping. That is, it is a transitional state and is either starting up or shutting down. While in this temporary state, it is not eligible for curtailment. |

|  |  |  |
| --- | --- | --- |
| releaseQueue |  | The load is ineligible for curtailment. The load has completed its curtailment and is presently timing down the restore time before it is again eligible for curtailment. |

|  |  |  |
| --- | --- | --- |
| unlocked |  | The load has recently been unlocked. It will analyze all conditions and set its present eligibility state after analysis completes. |

## **Scenario**

This section presents the experimental scenario for Transactive Energy simulations. This canonical experiment scenario provides for the orchestration of a transactive energy experiment.

It provides that, for any set of transactive energy components, that are based on the models of this document, a common experiment engine can be run which will produce the results of the simulation that can be compared.

It assumes the following:

1. The platform on which the experiment is running has a class model based on the Abstract Component Model;
2. The interfaces of the model are implemented as publish-subscribe messaging pattern or an equivalent; and
3. The interfaces provide at least the specified data from the component model interface definitions.

Note that whether the model components can be combined from different sources on any given platform can't be guaranteed by the abstract model. But if they are compatible the experiments can be composed.

### **Base TE Experiment Scenario diagram**

The base scenario for TE experiments is in Figure 6 contains the following:

1. Initialization of all components by the Experiment Manager; and
2. Three parallel sequences that continue until experiment ends:

* Physical: represents the timing needed to perform multiphysics power and energy simulation;
* Logical Controller: represents the timing needed to perform supervisory and local control of resources; and
* Transactive: represents the timing needed to perform a periodic transactive sequence resulting in a pricing model for the duration of the transactive step. This includes a "settle" loop for performing market/participant convergence on price.

Figure 6 illustrates the data flows and the target destinations of data for those components to use. In implementing this model, pub-sub or request response can produce equivalent results and the arrows and data flowed interpreted appropriately to the message mechanism.



Base TE Experiment Scenario

### **Settle**

The Settle loop allows for transactive pricing to settle during a negotiation for a single iteration of transactive negotiations.

Settle Detail

### **TE experiment loop**

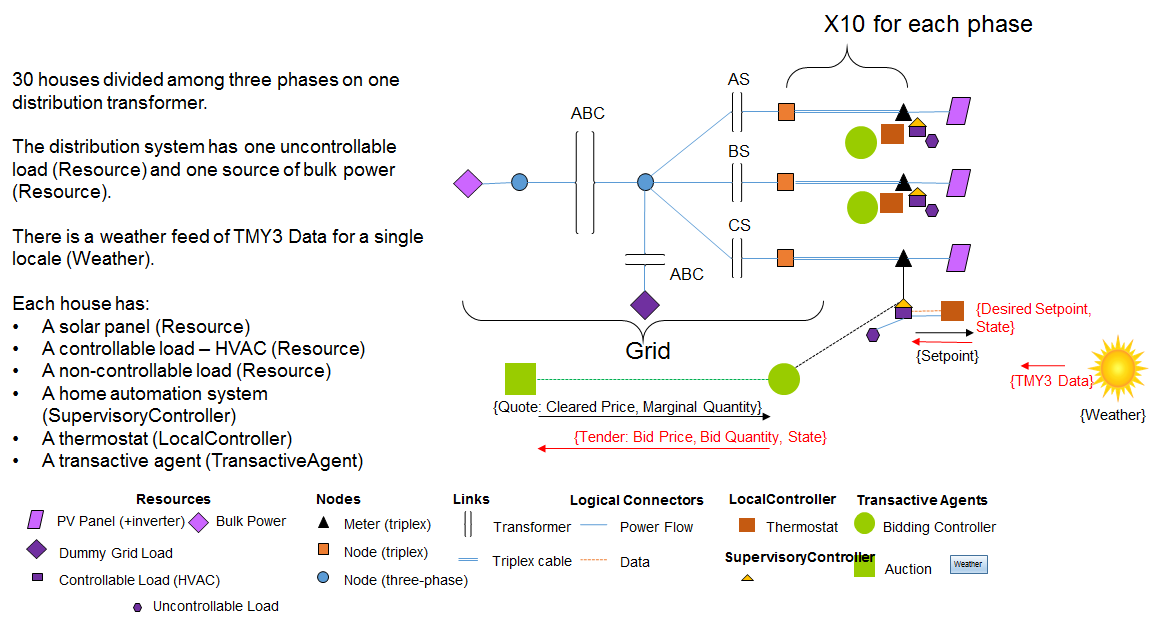
This fragment comprises the main experimental execution sequence. It has the three parallel timing sections.

TE experiment loop Detail

## **Beta Use Case**

A BetaUseCase was provided by PNNL contributors to this modeling effort. The use case itself was realized completely within Gridlab-D and is available from: <https://github.com/usnistgov/TEChallengeComponentModel>.

Figure 7 illustrates the 30 house "beta use case" scenario implemented entirely in GridLAB-D. It presents a minimal distribution segment that exposes three single phase transformers each feeding ten (10) houses. The houses are composed of an HVAC simulation, various loads, and the ability to bid into a transactive energy double-auction model.



1. Beta Use Case

The balance of this section will illustrate the beta use case and its components, including the traceability to TEM. The TEM model exclusively uses SI units. However, GridLAB-D uses predominantly Imperial units.

### **Beta Use Case diagram**

The Beta Use Case is a scenario chosen to illustrate the interactions of the components of the TE Abstract Component Model. Since its main goal is to illustrate the workings of the TE components and provide a reference simulation to use in developing more realistic and useful use cases, it should be considered in that context.

Figure 7 shows a UML package diagram that is a map to how the various parts of the use case are illustrated.

The House models, bid into a common Auction. All receive a common set of weather conditions.

Separate diagrams are presented to contain the Gridlab-D models of the Grid, PhaseAHouse, PhaseBHouse, and PhaseCHouse. In each of these diagrams, object instances based on the classes from the Common Component Inheritance diagram are arranged with their initial configuration parameters set. One example of each phase-connected house is provided.



Beta Use Case

### **Common Component Inheritance from Gridlab-D**

This section describes the specific classes derived from the Abstract Component Model components and associated additional data types used in the simulation of the Beta Use Case scenario. These derived classes substantially match the interfaces of the Gridlab-D model components used in the simulation. Note that the attributes of these model components are a subset of those available from the Gridlab-D models but represent those that were set as parameters in the beta use case simulation.

#### **Common Component Inheritance diagram**

Figure 9 shows an inheritance model of the Gridlab-D model classes mapped to Abstract Component model.

Note that some parts of the Gridlab-D classes are not seperate and distinct classes but do correspond to the functions of the component model. In these cases, pseudo classes are identified in the diagram (in yellow -- Null\_Controller, ZIPLoad\_controller, and HVAC\_controller) to make the inheritance map complete.

Additionally, several Gridlab-D classes are used in composing the Grid and are not represented uniquely in the component model. These are included in this diagram for reference but do not show inheritance from the component model classes.



Common Component Inheritance

#### **Auction**

A model that implements a double-auction bid resolution algorithm.

Auction Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| current\_price\_mean\_24h | double | Current price mean. |

|  |  |  |
| --- | --- | --- |
| current\_price\_stdev\_24h | double | Current price standard deviation. |

#### **BillDump**

A model that produces a summary electric bill for each metered house.

BillDump Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| filename | char | File name. |

|  |  |  |
| --- | --- | --- |
| group | char | Group name. |

|  |  |  |
| --- | --- | --- |
| runtime | char | Elapsed run time. |

#### **Climate**

A model of the weather.

Climate Detail

#### **Clock**

Represents the time basis of the experiment.

Clock Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| starttime | char | The start time global variable determines the time at which the simulation begins. Until the clock is set to the start time after the initialization is completed the clock is set to INIT. Before the first iteration begins, the clock will be set to start time. |

|  |  |  |
| --- | --- | --- |
| stoptime | char | The stoptime global variable determine the time at which the simulation ends. By default stoptime is set to NEVER, which is used to indicate that the simulation should stop only when it has reached a steady state. |

|  |  |  |
| --- | --- | --- |
| timezone | char | The timezone clock directive determines which timezone to use during the simulation. The timezone must be known before any timestamps can be interpreted. The timezone rules are used to determine the offset from UTC for all time calculations, as well as determine daylight or summer time shifts. |

#### **GroupRecorder**

A recorder that produces several outputs as Comma-Separated Values (CSV) files.

GroupRecorder Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| group | char | Group definition of common attributes to output. |

#### **House**

A model of a house with its loads and solar generation.

House Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| air\_temperature | double | Indoor air temperature. Unit: degF |

|  |  |  |
| --- | --- | --- |
| airchange\_per\_hour | double | Number of air-changes per hour. |

|  |  |  |
| --- | --- | --- |
| cooling\_COP | double | System cooling performance coefficient. Unit: Btu/kWh |

|  |  |  |
| --- | --- | --- |
| cooling\_system\_type | CoolingSystemType | This attribute defines cooling system types. |

|  |  |  |
| --- | --- | --- |
| fan\_type | FanType | Circulation fan (TWO\_SPEED, ONE\_SPEED, NONE) |

|  |  |  |
| --- | --- | --- |
| floor\_area | double | Home conditioned floor area. Unit: sf |

|  |  |  |
| --- | --- | --- |
| heating\_setpoint | double | Thermostat heating setpoint. Unit: degF |

|  |  |  |
| --- | --- | --- |
| heating\_system\_type | HeatingSystemType | Heating system type. |

|  |  |  |
| --- | --- | --- |
| hvac\_breaker\_rating | double | Determines the amount of current the HVAC circuit breaker can handle. Unit: A |

|  |  |  |
| --- | --- | --- |
| hvac\_power\_factor | double | Power factor of HVAC. Unit: dimensionless |

|  |  |  |
| --- | --- | --- |
| mass\_temperature | double | Interior mass temperature. Unit: degF |

|  |  |  |
| --- | --- | --- |
| motor\_efficiency | MotorEfficiency | Describes efficiency of the motor when using a motor model (VERY\_GOOD, GOOD, AVERAGE, POOR, VERY\_POOR) |

|  |  |  |
| --- | --- | --- |
| motor\_model | MotorModel | Indicates the level of detail used in modeling the HVAC motor parameters (FULL, BASIC, NONE) |

|  |  |  |
| --- | --- | --- |
| name | char | The name of house. |

|  |  |  |
| --- | --- | --- |
| number\_of\_doors | int | Ratio of door area to wall area. |

|  |  |  |
| --- | --- | --- |
| parent | char | Name of parent meter object this house is attached to. |

|  |  |  |
| --- | --- | --- |
| Rdoors | int | Door R-value. Unit: degF.sf.h/Btu |

|  |  |  |
| --- | --- | --- |
| Rfloor | double | Floor R-value. Unit: degF.sf.h/Btu |

|  |  |  |
| --- | --- | --- |
| Rroof | double | Roof R-value. Unit: degF.sf.h/Btu |

|  |  |  |
| --- | --- | --- |
| Rwall | double | Wall R-value. Unit: degF.sf.h/Btu |

|  |  |  |
| --- | --- | --- |
| Rwindows | double | Window R-value. Unit: degF.sf.h/Btu |

|  |  |  |
| --- | --- | --- |
| schedule\_skew | int | Seconds to skew scheduled times by. |

|  |  |  |
| --- | --- | --- |
| total\_thermal\_mass\_per\_floor\_area | int | Total thermal mass per floor area. Unit: Btu/degF.sf |

#### **HVAC\_Controller**

A Gridlab-D HVAC system "controller".

HVAC\_Controller Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| average\_target | char | This value points to the property within the auction object which will be used to provide the rolling average price. This is usually determined by a rolling 24 hour average (avg24), a rolling 3-day (avg72), or a rolling week (avg168). Future implementations will allow this rolling average to be determined at any window length. Future implementations will also include the ability to look at variables other than average price and standard deviation. Unit:Property identifier. |

|  |  |  |
| --- | --- | --- |
| base\_setpoint | char | This is the temperature set point of the system were there no controller present, or the original set point prior to the controller's input. Future implementations will allow this to control set points other than the temperature. No limit to value. Heating and cooling versions are used in the double\_ramp mode. Unit: DegF |

|  |  |  |
| --- | --- | --- |
| control\_mode | ControlMode | This specifies between the various control modes available. These will be further described in the specification documentation.   * RAMP * DOUBLE\_RAMP |

|  |  |  |
| --- | --- | --- |
| deadband | char | This is used to point the object property that contains the deadband variable. This is used in DEADBAND resolve\_mode. Unit:Property |

|  |  |  |
| --- | --- | --- |
| demand | char | The property name within the parent object that specifies the amount of power demanded by the controllable object at that time. For HVAC systems, this is heating\_demand or cooling\_demand. Unit: Property |

|  |  |  |
| --- | --- | --- |
| load | HVAC\_Load | The property name within the parent object that specifies the amount of power actually being used by the controllable object at the specified time. For HVAC systems, this is hvac\_load. |

|  |  |  |
| --- | --- | --- |
| market | char | This references the market that provides the price signal to the controller, and generates the rolling average and standard deviations seen by the object. This is also the object into which the controller will bid its price. It is typically specified as an auction or stubauction object, and is typically referenced by the name of the object. |

|  |  |  |
| --- | --- | --- |
| name | char | This is a name of HVAC\_Controller. |

|  |  |  |
| --- | --- | --- |
| period | int | The period of time for which the controller operates. This signals how often the controller will update the state of the set point and how often the controller will bid into the market. Ideally, this should be identical to, or a multiple of, the auction object’s time period. While this is not required, if the supply bid and demand bids do not coincide, odd behavior may occur. Must be a positive, non-zero value. Unit:Second |

|  |  |  |
| --- | --- | --- |
| ramp\_high | int | This specifies the slope of the linear control algorithm as a function of the average price, the current price, and the standard deviation from the average, and determines the controllers operation and bid. No limit to value. Heating and cooling versions are used in the double\_ramp mode. Unit:degF |

|  |  |  |
| --- | --- | --- |
| ramp\_low | int | This specifies the slope of the linear control algorithm as a function of the average price, the current price, and the standard deviation from the average, and determines the controllers operation and bid. No limit to value. Heating and cooling versions are used in the double\_ramp mode. Unit:degF |

|  |  |  |
| --- | --- | --- |
| range\_high | int | These are the maximum bounds of variability allowed by the controller. For example, the heating\_setpoint may vary +/- 5 degrees F, but no more. These are relative to the base\_setpoint (+5 F), not absolute (72 F). Range\_high must be zero or greater and range\_low must be zero or less. Heating and cooling versions are used in the double\_ramp mode |

|  |  |  |
| --- | --- | --- |
| range\_low | int | These are the maximum bounds of variability allowed by the controller. For example, the heating\_setpoint may vary +/- 5 degrees, but no more. These are relative to the base\_setpoint (+5 F), not absolute (72 F). Range\_high must be zero or greater and range\_low must be zero or less. Heating and cooling versions are used in the double\_ramp mode |

|  |  |  |
| --- | --- | --- |
| setpoint | char | The name of the set point to be modified by the controller object. Within the HVAC system, this would include heating\_setpoint or cooling\_setpoint. Heating and cooling versions of variable are used in DOUBLE\_RAMP mode. |

|  |  |  |
| --- | --- | --- |
| standard\_deviation\_target | char | Similar to average\_target, but specifies the rolling standard deviation. Unit:Property |

|  |  |  |
| --- | --- | --- |
| state | char | The property name within the parent object that specifies the current conditional state of the controllable object. For the HVAC system, this signifies on or off, however, future implementations may include multi-state objects. Unit:Property |

|  |  |  |
| --- | --- | --- |
| target | char | This determines the property within the parent object that is observed by the controller, and in conjunction with the set point property, is used to determine the bid of the parent object. Within the HVAC system, this would be the air\_temperature. Unit: Property |

|  |  |  |
| --- | --- | --- |
| total | char | The property name within the parent object that specifies, if any, all uncontrollable loads within that object in addition to the controllable load. For the HVAC model, this includes such things as circulation fan power or standby power settings, and is specified with total\_load. It does not include additional panel demand from other appliances. Unit:Property |

|  |  |  |
| --- | --- | --- |
| use\_predictive\_bidding | boolean | use predictive bidding. |

#### **HVAC\_Load**

This is an HVAC load.

HVAC\_Load Detail

#### **Inverter**

An inverter model.

Inverter Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| four\_guadrant\_control\_mode | char | Defines the type of inverter technology and the efficiency of the unit. NOTE: efficiency needs to be made a variable.   * TWO\_PULSE * SIX\_PULSE * TWELVE\_PULSE * PWM   (Unit: N/A) |

|  |  |  |
| --- | --- | --- |
| generator\_status | GeneratorStatus | Allows user to define when the generator is in operation or not.   * ONLINE * OFFLINE |

|  |  |  |
| --- | --- | --- |
| inverter\_efficiency | double | Efficiency of the inverter. This is assigned by inverter\_type and cannot be overridden at this time. Unit: unit |

|  |  |  |
| --- | --- | --- |
| inverter\_manufacturer | char | Name of manufacturer. |

|  |  |  |
| --- | --- | --- |
| inverter\_type | InverterType | Defines the type of inverter technology and the efficiency of the unit. NOTE: efficiency needs to be made a variable.   * TWO\_PULSE * SIX\_PULSE * TWELVE\_PULSE * PWM |

|  |  |  |
| --- | --- | --- |
| maximum\_dc\_power | double | This is max DC power. |

|  |  |  |
| --- | --- | --- |
| name | char | Name of this inverter instance. |

|  |  |  |
| --- | --- | --- |
| phase | Phases | Not used at this time – phases are assumed from the interconnection point. |

|  |  |  |
| --- | --- | --- |
| power\_factor | double | Defines the desired power factor in CONSTANT\_PF mode. Unit:unit |

|  |  |  |
| --- | --- | --- |
| rated\_power | double | This is rated power. |

|  |  |  |
| --- | --- | --- |
| solar | Solar | This is Solar type. |

|  |  |  |
| --- | --- | --- |
| use\_multipoint\_efficiency | boolean | This is use multipoint efficiency. |

#### **Meter**

A model of an electric meter.

Meter Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| bustype | BusType | This is a BusType. |

|  |  |  |
| --- | --- | --- |
| nominal\_voltage | double | This is nominal voltage. |

#### **Null\_Controller**

A "null" controller to help make the mapping complete. This model does not exist in Gridlab-D and has no expected behavior.

Null\_Controller Detail

#### **Player**

A model that "plays" data into the Gridlab-D simulation.

Player Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| file | char | File name. |

|  |  |  |
| --- | --- | --- |
| interpolate | char | Interpolation. |

|  |  |  |
| --- | --- | --- |
| loop | int | Number of loops. |

|  |  |  |
| --- | --- | --- |
| name | char | This is name of player. |

|  |  |  |
| --- | --- | --- |
| value | int | Value. |

#### **Recorder**

A model that allows formatted CSV output from the Gridlab-D simulation.

Recorder Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| file | char | File name. |

|  |  |  |
| --- | --- | --- |
| interval | int | Time interval, such as 60 seconds. |

|  |  |  |
| --- | --- | --- |
| parent | char | Parent object name. |

|  |  |  |
| --- | --- | --- |
| property | char | Some object properties, such as temperature, humidity,.... |

#### **Solar**

A solar energy model.

Solar Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| generator\_mode | GeneratorMode | Only operational in SUPPLY\_DRIVEN at this time.   * UNKNOWN * CONSTANT\_V * CONSTANT\_PQ * CONSTANT\_PF * SUPPLY\_DRIVEN   (Unit: N/A) |

|  |  |  |
| --- | --- | --- |
| generator\_status | GeneratorStatus | Default is ONLINE. Allows a user to dropout a generator.   * OFFLINE * ONLINE   (Unit: N/A) |

|  |  |  |
| --- | --- | --- |
| orientation | char | Orientation of solar array. |

|  |  |  |
| --- | --- | --- |
| panel\_type | PanelType | Uses pre-defined panel technologies. Defines efficiency, Pmax\_temp\_coeff, and Voc\_temp\_coeff.   * SINGLE\_CRYSTAL\_SILICON (default) * MULTI\_CRYSTAL\_SILICON * AMORPHOUS\_SILICON * THIN\_FILM\_GA\_AS (incomplete) * CONCENTRATOR (incomplete)   (Unit: N/A) |

|  |  |  |
| --- | --- | --- |
| rated\_power | double | Rated power. |

#### **Transformer**

Transformers are derived from the link class and inherit all of its properties.

Transformer Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| configuration | Transformer\_configuration | Transformer\_configuration object that describes the specific transformer implementation. Type: object Unit: N/A |

|  |  |  |
| --- | --- | --- |
| from | char | One connecting end of the link object. This will be the name or reference to a node-based object elsewhere in the powerflow model. Type: object (char: object Id) Unit: N/A |

|  |  |  |
| --- | --- | --- |
| groupid | char | This is a groupId. |

|  |  |  |
| --- | --- | --- |
| inputId | GridNodeId | Input Ids. |

|  |  |  |
| --- | --- | --- |
| inputs | Phases | Inputs. |

|  |  |  |
| --- | --- | --- |
| name | char | Name of transformer instance. |

|  |  |  |
| --- | --- | --- |
| outputId | GridNodeId | Output Id. |

|  |  |  |
| --- | --- | --- |
| outputs | Phases | Outputs. |

|  |  |  |
| --- | --- | --- |
| phases | Phases | This is a phase. |

|  |  |  |
| --- | --- | --- |
| to | char | The other connecting end of the link object. This will be the name or reference to a node-based object elsewhere in the power flow model. (Unit:N/A) |

#### **Transformer\_configuration**

Model of a transformer configuration.

Transformer\_configuration Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| connect\_type | ConnectType | Describes the electrical connection between the high and low side of the transformer. These may be referenced by keyword or number 0 - UNKNOWN - An unknown transformer that will throw an error when used. 1 - WYE\_WYE - A wye to wye connected transformer. 2 - DELTA\_DELTA - A delta to delta connected transformer. 3 - DELTA\_GWYE - A delta to grounded-wye connected transformer. 4 - SINGLE\_PHASE - A single leg of a wye to wye connected transformer. 5 - SINGLE\_PHASE\_CENTER\_TAPPED - A single-phase, center-tapped transformer or split-phase transformer. Used to connect three-phase distribution to triplex- distribution. |

|  |  |  |
| --- | --- | --- |
| impedance | Impedance | De-referenced characteristic impedance of the transformer. Note that resistance and reactance above directly write the real and complex portions of this parameter, so only resistance and reactance or just impedance need to be specified. |

|  |  |  |
| --- | --- | --- |
| install\_type | InstallType | Describes the type of transformer the object represents. Used for informational purposes only. Valid types may be referenced by keyword or number 0 - UNKNOWN - No information on the transformer physical type. 1 - POLETOP - A pole-mounted transformer. 2 - PADMOUNT - A pad, or ground level transformer. 3 - VAULT - An enclosed transformer "building," either underground or above ground. |

|  |  |  |
| --- | --- | --- |
| name | char | This is name of Transformer configuration. |

|  |  |  |
| --- | --- | --- |
| power\_rating | double | Nominal power rating of the entire transformer. Unit: kilo-Volt Amperes |

|  |  |  |
| --- | --- | --- |
| powerA\_rating | double | Nominal power rating of windings associated with phase A if wye-connected or AB if delta-connected. Unit: kilo-Volt Amperes |

|  |  |  |
| --- | --- | --- |
| powerB\_rating | double | Nominal power rating of windings associated with phase B if wye-connected or BC if delta-connected. Unit: kilo-Volt Amperes |

|  |  |  |
| --- | --- | --- |
| powerC\_rating | double | Nominal power rating of windings associated with phase A if wye-connected or AB if delta-connected. Unit: kilo-Volt Amperes |

|  |  |  |
| --- | --- | --- |
| primary\_voltage | double | Nominal voltage of the primary winding side of the transformer. Unit: Volts. |

|  |  |  |
| --- | --- | --- |
| secondary\_voltage | double | Nominal voltage of the secondary winding side of the transformer. Unit: Volts. |

#### **Triplex\_line**

A triplex line model.

Triplex\_line Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| configuration | Triplex\_line\_configuration | triplex line configuration. |

|  |  |  |
| --- | --- | --- |
| groupId | char | Group Id. |

#### **Triplex\_line\_conductor**

A triplex line conductor model.

Triplex\_line\_conductor Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| geometric\_mean\_radius | double | GMR of conductor. Unit: feet |

|  |  |  |
| --- | --- | --- |
| name | char | This is name of Triplex line conductor. |

|  |  |  |
| --- | --- | --- |
| resistance | double | Resistance of the conductor. Unit: Ohm/mile |

#### **Triplex\_line\_configuration**

A triplex line configuration model.

Triplex\_line\_configuration Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| conductor\_1 | triplex\_line\_conductor | triplex\_conductor object that represents the physical wire of phase 1.  Unit:N/A |

|  |  |  |
| --- | --- | --- |
| conductor\_2 | triplex\_line\_conductor | triplex\_conductor object that represents the physical wire of phase 2.  Unit:N/A |

|  |  |  |
| --- | --- | --- |
| conductor\_3 | triplex\_line\_conductor | triplex\_conductor object that represents the physical wire of phase 3.  Unit:N/A |

|  |  |  |
| --- | --- | --- |
| diameter | double | Diameter of the conductor.  Unit:inches |

|  |  |  |
| --- | --- | --- |
| insulation\_thickness | double | Thickness of the insulation around the phase 1 and phase 2 conductors .  Unit:inches |

|  |  |  |
| --- | --- | --- |
| name | char | This is name of Triple line configuration. |

#### **Triplex\_meter**

A meter model of a triplex line.

Triplex\_meter Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| bill\_day | int | Sets the date of the month at which the final monthly bill is calculated (at midnight). Maximum value is 28. Unit: N/A |

|  |  |  |
| --- | --- | --- |
| bill\_mode | BillMode | Describes the method in which the meter receives its price signal. 0 - NONE - Billing is not used (default). 1 - UNIFORM - A static price is used through variable price, however, this may change over time using a player or schedule. 2 - TIERED - Tiered pricing plan where the price changes as a function of the amount of energy used in the month. See tier\_price and tier\_energy. 3 - HOURLY - This is used in conjunction with an auction or stubauction object. Receives its price directly from a market signal, but only updates on an hourly basis. Used in conjunction with power\_market. NOTE: while this says "hourly", it will actually update any time the price changes in the auction. 4 - TIERED\_RTP - Merges TIERED and HOURLY modes. It applies both a real time price via the auction to energy usage, but then also applies block / tiered rates to the total monthly energy use. (Unit: N/A) |

|  |  |  |
| --- | --- | --- |
| first\_tier\_energy | double | Determines the point at which the price of energy changes from price to first\_tier\_price.  Unit: kWh |

|  |  |  |
| --- | --- | --- |
| first\_tier\_price | char | When using bill\_mode TIERED, this determines the energy price after energy increases above first\_tier\_energy, but below second\_tier\_energy. If second\_tier\_energy is not defined, then this price will be used to infinity. While energy is below first\_tier\_energy, price is used to calculate the monthly\_bill.  Unit: $/kWh |

|  |  |  |
| --- | --- | --- |
| groupid | char | This is group id. |

|  |  |  |
| --- | --- | --- |
| invertor | Inverter | Inverter. |

|  |  |  |
| --- | --- | --- |
| monthly\_fee | double | This is a recurrent monthly service charge that is added into the bill on the first day of the billing cycle (no pro-rating). Unit: $. |

|  |  |  |
| --- | --- | --- |
| parent | char | Parent meter instance name. |

|  |  |  |
| --- | --- | --- |
| price\_base | char | Price base. |

|  |  |  |
| --- | --- | --- |
| second\_tier\_energy | double | Determines the point at which the price of energy changes from first\_tier\_price to second\_tier\_price. Unit: kWh |

|  |  |  |
| --- | --- | --- |
| second\_tier\_price | char | When using bill\_mode TIERED, this determines the energy price after energy increases above second\_tier\_energy, but below third\_tier\_energy. If third\_tier\_energy is not defined, then this price will be used to infinity. Unit: $/kWh. |

|  |  |  |
| --- | --- | --- |
| third\_tier\_energy | double | Determines the point at which the price of energy changes from second\_tier\_price to third\_tier\_price. Unit: kWh. |

|  |  |  |
| --- | --- | --- |
| third\_tier\_price | char | When using bill\_mode TIERED, this determines the energy price after energy increases above third\_tier\_energy and is used to infinite energy. Unit: $/kWh. |

#### **Unresp\_load**

Model of an unresponsive load.

Unresp\_load Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| base\_power\_A | double | In a format similar to ZIPload & load this represents the nominal power on phase A before applying ZIP fractions. Unit: VA |

|  |  |  |
| --- | --- | --- |
| base\_power\_B | double | In a format similar to ZIPload & load this represents the nominal power on phase B before applying ZIP fractions. Unit: VA |

|  |  |  |
| --- | --- | --- |
| base\_power\_C | double | In a format similar to ZIPload & load this represents the nominal power on phase C before applying ZIP fractions. Unit: VA |

|  |  |  |
| --- | --- | --- |
| current\_fraction\_A | double | This is the constant current fraction of base power on phase A. Unit: pu |

|  |  |  |
| --- | --- | --- |
| current\_fraction\_B | double | This is the constant current fraction of base power on phase B. Unit: pu |

|  |  |  |
| --- | --- | --- |
| current\_fraction\_C | double | This is the constant current fraction of base power on phase C. Unit: pu |

|  |  |  |
| --- | --- | --- |
| impedance\_fraction\_A | double | This is the constant impedance fraction of base power on phase A. Unit: pu |

|  |  |  |
| --- | --- | --- |
| impedance\_fraction\_B | double | This is the constant impedance fraction of base power on phase B. Unit: pu |

|  |  |  |
| --- | --- | --- |
| impedance\_fraction\_C | double | This is the constant impedance fraction of base power on phase C. Unit: pu |

|  |  |  |
| --- | --- | --- |
| nominal\_voltage | int | Nominal voltage. |

|  |  |  |
| --- | --- | --- |
| parent | char | Name of parent instance node where load is connected. |

|  |  |  |
| --- | --- | --- |
| power\_fraction\_A | double | This is the constant power fraction of base power on phase A. Unit:pu |

|  |  |  |
| --- | --- | --- |
| power\_fraction\_B | double | This is the constant power fraction of base power on phase B. Unit:pu |

|  |  |  |
| --- | --- | --- |
| power\_fraction\_C | double | This is the constant power fraction of base power on phase C. Unit:pu |

|  |  |  |
| --- | --- | --- |
| power\_pf\_A | double | In similar format as ZIPload & load this is the power factor of the phase A constant power portion of load. Unit: pu |

|  |  |  |
| --- | --- | --- |
| power\_pf\_B | double | In similar format as ZIPload & load this is the power factor of the phase B constant power portion of load. Unit: pu |

|  |  |  |
| --- | --- | --- |
| power\_pf\_C | double | In similar format as ZIPload & load this is the power factor of the phase C constant power portion of load. Unit: pu |

#### **ZIPLoad**

A zip load model. The ZIPload model uses a classic ZIP load model (constant impedance, current, and power) where "base power" is specified, then the ZIP fractions and power factors are assigned.

ZIPLoad Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| base\_power | double | This represents the nominal power on phase before applying ZIP fractions. Unit: VA |

|  |  |  |
| --- | --- | --- |
| current\_fraction | double | This is the constant current fraction of base power on phase. Unit:pu |

|  |  |  |
| --- | --- | --- |
| current\_pf | double | This is the power factor of the phase constant current portion of load. Unit:pu |

|  |  |  |
| --- | --- | --- |
| heat\_fraction | double | Heat fraction. |

|  |  |  |
| --- | --- | --- |
| impedance\_fraction | double | This is the constant impedance fraction of base power on phase . Unit:pu |

|  |  |  |
| --- | --- | --- |
| impedance\_pf | double | In similar format as ZIPload this is the power factor of the phase constant impedance portion of load. Unit:pu |

|  |  |  |
| --- | --- | --- |
| power\_fraction | double | This is the constant power fraction of base power on phase. Unit:pu |

|  |  |  |
| --- | --- | --- |
| power\_pf | double | In similar format as ZIPload this is the power factor of the phase constant power portion of load. Unit: pu |

|  |  |  |
| --- | --- | --- |
| schedule\_skew | int | Schedule screw. |

#### **ZIPLoad\_controller**

A zip load model controller.

ZIPLoad\_controller Detail

### 

### **Grid**

This package describes the beta use case grid model.

#### **Grid-Part 1 diagram**

This diagram illustrates the beta use case grid model. This is part 1 of 2.



Grid-Part 1

#### **Grid Part 2 diagram**

This diagram illustrates the beta use case grid model. This is part 2 of 2.



Grid Part 2

#### **bulkPower**

bulkPower Detail

#### **bulkPower**

bulkPower Detail

#### **F1\_center\_tap\_transformer\_A**

F1\_center\_tap\_transformer\_A Detail

#### **F1\_center\_tap\_transformer\_A**

F1\_center\_tap\_transformer\_A Detail

#### **F1\_center\_tap\_transformer\_B**

F1\_center\_tap\_transformer\_B Detail

#### **F1\_center\_tap\_transformer\_B**

F1\_center\_tap\_transformer\_B Detail

#### **F1\_center\_tap\_transformer\_C**

F1\_center\_tap\_transformer\_C Detail

#### **F1\_center\_tap\_transformer\_C**

F1\_center\_tap\_transformer\_C Detail

#### **F1\_tpm\_flatrate\_A5**

F1\_tpm\_flatrate\_A5 Detail

#### **F1\_tpm\_flatrate\_A5**

F1\_tpm\_flatrate\_A5 Detail

#### **F1\_tpm\_flatrate\_B0**

F1\_tpm\_flatrate\_B0 Detail

#### **F1\_tpm\_flatrate\_B0**

F1\_tpm\_flatrate\_B0 Detail

#### **F1\_tpm\_flatrate\_C1**

F1\_tpm\_flatrate\_C1 Detail

#### **F1\_tpm\_flatrate\_C1**

F1\_tpm\_flatrate\_C1 Detail

#### **F1\_tpm\_rt\_A5**

F1\_tpm\_rt\_A5 Detail

#### **F1\_tpm\_rt\_B0**

F1\_tpm\_rt\_B0 Detail

#### **F1\_tpm\_rt\_C1**

F1\_tpm\_rt\_C1 Detail

#### **F1\_transformer\_meter**

F1\_transformer\_meter Detail

#### **F1\_transformer\_meter**

F1\_transformer\_meter Detail

#### **F1\_Transformer1**

F1\_Transformer1 Detail

#### **F1\_Transformer1**

F1\_Transformer1 Detail

#### **F1\_Triplex\_Line\_A5**

F1\_Triplex\_Line\_A5 Detail

#### **F1\_Triplex\_Line\_A5**

F1\_Triplex\_Line\_A5 Detail

#### **F1\_Triplex\_Line\_B0**

F1\_Triplex\_Line\_B0 Detail

#### **F1\_Triplex\_Line\_B0**

F1\_Triplex\_Line\_B0 Detail

#### **F1\_Triplex\_Line\_C1**

F1\_Triplex\_Line\_C1 Detail

#### **F1\_Triplex\_Line\_C1**

F1\_Triplex\_Line\_C1 Detail

#### **F1\_triplex\_node\_A**

F1\_triplex\_node\_A Detail

#### **F1\_triplex\_node\_A**

F1\_triplex\_node\_A Detail

#### **F1\_triplex\_node\_B**

F1\_triplex\_node\_B Detail

#### **F1\_triplex\_node\_B**

F1\_triplex\_node\_B Detail

#### **F1\_triplex\_node\_C**

F1\_triplex\_node\_C Detail

#### **F1\_triplex\_node\_C**

F1\_triplex\_node\_C Detail

#### **Link01**

Link01 Detail

#### **Link03**

Link03 Detail

#### **Link04**

Link04 Detail

#### **Link05**

Link05 Detail

#### **Name\_1\_0\_AA\_triplex**

Name\_1\_0\_AA\_triplex Detail

#### **Name\_1\_0\_AA\_triplex**

Name\_1\_0\_AA\_triplex Detail

#### **PHASE\_A**

PHASE\_A Detail

#### **PHASE\_A**

PHASE\_A Detail

#### **PHASE\_B**

PHASE\_B Detail

#### **PHASE\_B**

PHASE\_B Detail

#### **PHASE\_C**

PHASE\_C Detail

#### **PHASE\_C**

PHASE\_C Detail

#### **substation\_root**

substation\_root is a meter between bulk power and F1\_Transformer1, which is three phase node with 69000 V.

substation\_root Detail

#### **substation\_root**

substation\_root is a meter between bulk power and F1\_Transformer1, which is three phase node with 69000 V.

substation\_root Detail

#### **substation\_transformer\_configuration**

substation\_transformer\_configuration Detail

#### **substation\_transformer\_configuration**

substation\_transformer\_configuration Detail

#### **TLCFG**

TLCFG Detail

#### **TLCFG**

TLCFG Detail

#### **transformer\_A\_configuration**

transformer\_A\_configuration Detail

#### **transformer\_A\_configuration**

transformer\_A\_configuration Detail

#### **transformer\_B\_configuration**

transformer\_B\_configuration Detail

#### **transformer\_B\_configuration**

transformer\_B\_configuration Detail

#### **transformer\_C\_configuration**

transformer\_C\_configuration Detail

#### **transformer\_C\_configuration**

transformer\_C\_configuration Detail

#### **unresp\_load**

unresp\_load Detail

#### **unresp\_load**

unresp\_load Detail

### 

### **PhaseAHouse**

This package describes the beta use case model for a house on Phase A. Note that there are 10 such houses and only one is shown.

#### **PhaseAHouse diagram**

Houses connected to Phase A are shown in this diagram.



PhaseAHouse

#### **clothes\_Washer\_A5**

clothes\_Washer\_A5 Detail

#### **dryer\_A5**

dryer\_A5 Detail

#### **F1\_house\_A5**

F1\_house\_A5 Detail

#### **hvac\_Controller\_F1\_House\_A5**

hvac\_Controller\_F1\_House\_A5 Detail

#### **hvac\_Load\_A5**

hvac\_Load\_A5 Detail

#### **inv\_F1\_house\_A5**

inv\_F1\_house\_A5 Detail

#### **lights\_A5**

lights\_A5 Detail

#### **microwave\_A5**

microwave\_A5 Detail

#### **range\_A5**

range\_A5 Detail

#### **refrigerator\_A5**

refrigerator\_A5 Detail

#### **sol\_inv\_F1\_house\_A5**

sol\_inv\_F1\_house\_A5 Detail

#### **solar\_F1\_tpm\_rt\_A5**

solar\_F1\_tpm\_rt\_A5 Detail

#### **zipLoad\_Controller\_F1\_House\_A5**

zipLoad\_Controller\_F1\_House\_A5 Detail

### 

### **PhaseBHouse**

This package describes the beta use case model for a house on Phase B. Note that there are 10 such houses and only one is shown.

#### **PhaseBHouse diagram**

Houses connected to Phase B are shown in this diagram.



PhaseBHouse

#### **clothes\_Washer\_B0**

clothes\_Washer\_B0 Detail

#### **dryer\_B0**

dryer\_B0 Detail

#### **F1\_house\_B0**

F1\_house\_B0 Detail

#### **hvac\_Controller\_F1\_House\_B0**

hvac\_Controller\_F1\_House\_B0 Detail

#### **hvac\_Load\_B0**

hvac\_Load\_B0 Detail

#### **inv\_F1\_house\_B0**

inv\_F1\_house\_B0 Detail

#### **lights\_B0**

lights\_B0 Detail

#### **microwave\_B0**

microwave\_B0 Detail

#### **range\_B0**

range\_B0 Detail

#### **refrigerator\_B0**

refrigerator\_B0 Detail

#### **sol\_inv\_F1\_house\_B0**

sol\_inv\_F1\_house\_B0 Detail

#### **solar\_F1\_tpm\_rt\_B0**

solar\_F1\_tpm\_rt\_B0 Detail

#### **zipLoad\_Controller\_F1\_House\_B0**

zipLoad\_Controller\_F1\_House\_B0 Detail

### 

### **PhaseCHouse**

This package describes the beta use case model for a house on Phase C. Note that there are 10 such houses and only one is shown.

#### **PhaseCHouse diagram**

Houses connected to Phase C are shown in this diagram.



PhaseCHouse

#### **clothes\_Washer\_C1**

clothes\_Washer\_C1 Detail

#### **dishwasher\_C1**

dishwasher\_C1 Detail

#### **dryer\_C1**

dryer\_C1 Detail

#### **F1\_house\_C1**

F1\_house\_C1 Detail

#### **F1\_house\_C1\_TA**

F1\_house\_C1\_TA Detail

#### **hvac\_Controller\_F1\_House\_C1**

hvac\_Controller\_F1\_House\_C1 Detail

#### **hvac\_Load\_C1**

hvac\_Load\_C1 Detail

#### **inv\_F1\_house\_C1**

inv\_F1\_house\_C1 Detail

#### **lights\_C1**

lights\_C1 Detail

#### **microwave\_C1**

microwave\_C1 Detail

#### **range\_C1**

range\_C1 Detail

#### **refrigerator\_C1**

refrigerator\_C1 Detail

#### **sol\_inv\_F1\_house\_C1**

sol\_inv\_F1\_house\_C1 Detail

#### **solar\_F1\_tpm\_rt\_C1**

solar\_F1\_tpm\_rt\_C1 Detail

#### **zipLoad\_Controller\_F1\_House\_C1**

zipLoad\_Controller\_F1\_House\_C1 Detail

### 

### **Gridlab-D Enumerations**

Enumerations used in the beta use case model.

#### **Enumerations diagram**

GridLAB-D Enumerations.



Enumerations

#### **BillMode**

BillMode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| NONE |  | 0 - NONE - Billing is not used (default). |

|  |  |  |
| --- | --- | --- |
| UNIFORM |  | 1 - UNIFORM - A static price is used through variable price, however, this may change over time using a player or schedule. |

|  |  |  |
| --- | --- | --- |
| HOURLY |  | 3 - HOURLY - This is used in conjunction with an auction or stubauction object. Receives its price directly from a market signal, but only updates on an hourly basis. Used in conjunction with power\_market. NOTE: while this says "hourly", it will actually update any time the price changes in the auction. |

|  |  |  |
| --- | --- | --- |
| TIERED\_RTP |  | 4 - TIERED\_RTP - Merges TIERED and HOURLY modes. Applies both a real time price via the auction to energy usage, but then also applies block / tiered rates to the total monthly energy use. |

#### **BusType**

BusType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| PQ |  | The PQ bus is the most commonly found bus type in electric network models. PQ buses are nodes where both the real power (P) and reactive power (Q) are given. |

|  |  |  |
| --- | --- | --- |
| PV |  | The PV bus is the next most common bus type in electric network models. PV buses are nodes where the real power (P) is given, but the reactive power (Q) must be determined at each iteration. |

|  |  |  |
| --- | --- | --- |
| SWING |  | The SWING bus occurs at least one in any given island of a network models. Large models may have more the one SWING bus, particularly if areas of the network are only lighting coupled by relatively high impedance links. SWING bus nodes are nodes where both the real power (P) and reactive power (Q) must be determined at each iteration. |

#### **ConnectType**

Describes the electrical connection between the high and low side of the transformer. These may be referenced by keyword or number

0 - UNKNOWN - An unknown transformer that will throw an error when used.

1 - WYE\_WYE - A wye to wye connected transformer.

2 - DELTA\_DELTA - A delta to delta connected transformer.

3 - DELTA\_GWYE - A delta to grounded-wye connected transformer.

ConnectType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| UNKNOWN |  | 0 - UNKNOWN - An unknown transformer that will throw an error when used. |

|  |  |  |
| --- | --- | --- |
| WYE\_WYE |  | 1 - WYE\_WYE - A wye to wye connected transformer. |

|  |  |  |
| --- | --- | --- |
| DELTA\_DELTA |  | 2 - DELTA\_DELTA - A delta to delta connected transformer. |

|  |  |  |
| --- | --- | --- |
| DELTA\_GWYE |  | 3 - DELTA\_GWYE - A delta to grounded-wye connected transformer. |

#### **ControlMode**

This specifies between the various control modes available. These will be further described in the specification documentation.

* RAMP
* DOUBLE\_RAMP

ControlMode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| RAMP |  | Ramp control mode. |

|  |  |  |
| --- | --- | --- |
| DOUBLE\_RAMP |  | Double\_ramp control mode. |

#### **CoolingSystemType**

Set cooling mechanism for house (HEAT\_PUMP, ELECTRIC, NONE)

CoolingSystemType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| HEAT\_PUMP |  | Heat pump cooling system. |

|  |  |  |
| --- | --- | --- |
| ELECTRIC |  | Electric cooling system. |

|  |  |  |
| --- | --- | --- |
| NONE |  | None cooling system. |

#### **FanType**

Circulation fan (TWO\_SPEED, ONE\_SPEED, NONE)

FanType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| TWO\_SPEED |  | Two speed fun. |

|  |  |  |
| --- | --- | --- |
| ONE\_SPEED |  | One speed fun. |

|  |  |  |
| --- | --- | --- |
| NONE |  | None fan. |

#### **GeneratorMode**

GeneratorMode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| UNKNOWN |  | Unknown mode. |

|  |  |  |
| --- | --- | --- |
| CONSTANT\_V |  | Constant\_V Mode. |

|  |  |  |
| --- | --- | --- |
| CONSTANT\_PQ |  | Constant\_PQ Mode. |

|  |  |  |
| --- | --- | --- |
| CONSTANT\_PF |  | Constant\_PF Mode |

|  |  |  |
| --- | --- | --- |
| SUPPLY\_DRIVEN |  | Supply\_driven mode. |

#### **GeneratorStatus**

GeneratorStatus Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| ONLINE |  | Online status. |

|  |  |  |
| --- | --- | --- |
| OFFLINE |  | Offline status. |

#### **HeatingSystemType**

Set heating mechanism for house (RESISTANCE, HEAT\_PUMP, GAS, NONE)

HeatingSystemType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| RESISTANCE |  | Resistance heating system. |

|  |  |  |
| --- | --- | --- |
| HEAT\_PUMP |  | Heat pump heating system. |

|  |  |  |
| --- | --- | --- |
| GAS |  | Gas heating system. |

|  |  |  |
| --- | --- | --- |
| NONE |  | None heating system. |

#### **InstallType**

Describes the type of transformer the object represents. Used for informational purposes only. Valid types may be referenced by keyword or number

0 - UNKNOWN - No information on the transformer physical type.

1 - POLETOP - A pole-mounted transformer.

2 - PADMOUNT - A pad, or ground level transformer.

3 - VAULT - An enclosed transformer "building," either underground or above ground.

InstallType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| UNKNOWN |  | 0 - UNKNOWN - No information on the transformer physical type. |

|  |  |  |
| --- | --- | --- |
| POLETOP |  | 1 - POLETOP - A pole-mounted transformer. |

|  |  |  |
| --- | --- | --- |
| PADMOUNT |  | 2 - PADMOUNT - A pad, or ground level transformer. |

|  |  |  |
| --- | --- | --- |
| VAULT |  | 3 - VAULT - An enclosed transformer "building," either underground or above ground. |

#### **InverterType**

InverterType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| TWO\_PULSE |  | Tow\_pulse mode. |

|  |  |  |
| --- | --- | --- |
| SIX\_PULSE |  | Six\_pulse mode. |

|  |  |  |
| --- | --- | --- |
| TWELVE\_PULSE |  | Twelve\_pulse mode. |

|  |  |  |
| --- | --- | --- |
| OWM |  | PWM type |

#### **MotorEfficiency**

Describes efficiency of the motor when using a motor model (VERY\_GOOD, GOOD, AVERAGE, POOR, VERY\_POOR)

MotorEfficiency Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| VERY\_GOOD |  | Very good efficiency. |

|  |  |  |
| --- | --- | --- |
| GOOD |  | Good efficiency. |

|  |  |  |
| --- | --- | --- |
| AVERAGE |  | Average efficiency. |

|  |  |  |
| --- | --- | --- |
| POOR |  | Poor efficiency. |

|  |  |  |
| --- | --- | --- |
| VERY\_POOR |  | Very poor efficiency. |

#### **MotorModel**

MotorModel Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| FULL |  | Full model. |

|  |  |  |
| --- | --- | --- |
| BASIC |  | Basic model. |

|  |  |  |
| --- | --- | --- |
| NONE |  | None model. |

#### **PanelType**

Uses pre-defined panel technologies. Defines efficiency, Pmax\_temp\_coeff, and Voc\_temp\_coeff.

PanelType Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| SINGLE\_CRYSTAL\_SILICON |  | Single\_crystal\_silicon panel type. |

|  |  |  |
| --- | --- | --- |
| MULTI\_CRYSTAL\_SILICON |  | Multi\_crystal\_silicon panel type. |

|  |  |  |
| --- | --- | --- |
| AMORPHOUS\_SILICON |  | Amorphous\_silicon panel type. |

|  |  |  |
| --- | --- | --- |
| THIN\_FILM\_GA\_AS |  | Thin\_film\_GA\_As panel type. |

|  |  |  |
| --- | --- | --- |
| CONCENTRATOR |  | Concentrator panel type. |

#### **PowerTypes**

Defines whether the connection is AC or DC. This variable is not currently used at this time as the connection method is determined from the connection device (meter vs. inverter).

* AC
* DC

PowerTypes Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| AC |  | Alternative current. |

|  |  |  |
| --- | --- | --- |
| DC |  | Directive current. |

#### **ResolveMode**

ResolveMode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| DEADBAND |  | Deadband mode. |

|  |  |  |
| --- | --- | --- |
| SLIDING |  | Sliding mode. |

#### **SystemMode**

Heating/cooling system operation state (UNKNOWN, HEAT, OFF, COOL, AUX)

SystemMode Detail

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| UNKNOWN |  | Unknown mode. |

|  |  |  |
| --- | --- | --- |
| HEAT |  | Heat mode. |

|  |  |  |
| --- | --- | --- |
| OFF |  | Off mode. |

|  |  |  |
| --- | --- | --- |
| COOL |  | Cool mode |

|  |  |  |
| --- | --- | --- |
| AUX |  | Aux mode. |

### **auction**

auction Detail

### **climate**

climate Detail

## **Composite and Extended Classes**

Composite classes allows for the composition and testing of classes that realize selected sets of interfaces. In any given instance of a TE Component, one or more of the roles or interfaces may be realized. Exchanges shown in the Base TE Experiment Scenario between components that have been combined do not occur during the experiment.

### **Composite and Extended Classes diagram**

This diagram illustrates how components can be combined and/or extended for use in experiments of the Abstract Component Model.



Composite and Extended Classes

### **AllInOne**

Contains Resource, Local Controller, Supervisory Controller with Transactive interface. This would represent the transactive appliance from Figure 1.

AllInOne Detail

### **ExtendedSupervisoryController**

This extended SupervisoryController exposes an additional CustomInterface. By substituting this version of the SupervisoryController additional capabilities can be exposed when substituting from the base.

ExtendedSupervisoryController Detail

### **ResourceWithLocalController**

Contains Resource with Local Controller.

ResourceWithLocalController Detail

### **CustomInterface**

A custom extension interface for one of the core components.

CustomInterface Detail