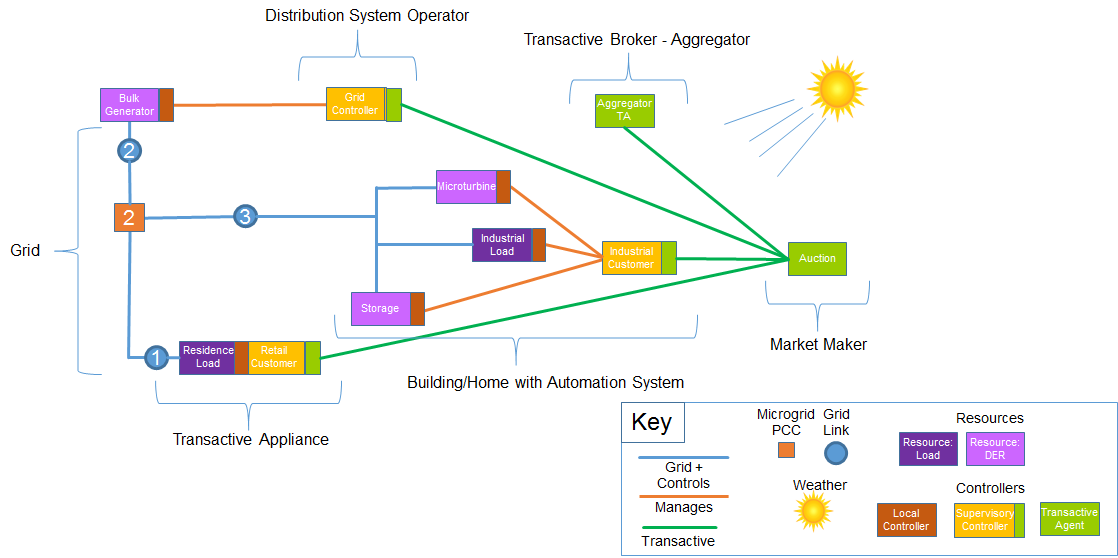
# **T****he Transactive Energy Challenge Abstract Component Model**

## **O****verview**

This model was the product of a "Tiger Team Effort" engaged by NIST, PNNL, Vanderbilt, and CMU/MIT during the summer of 2016 in support of the NIST Transactive Energy Challenge <https://pages.nist.gov/TEChallenge/>. 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 an Abstract Component Model, which provides a basis for common discussions, similar to the IEEE standard feeder models that the industry has used for more than decade to test changes to simulations. The feeder models have allowed apples to apples comparisons to happen, speeding up the development of better simulation tools, and understanding of modeling and simulation in most power engineering programs. The Extensible Component Model should be able to provide similar support for TE in the industry. A notional diagram of the model is below.



1. Notional Topology

While the diagram 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 SGIP TECG use cases (<http://www.sgip.org/wp-content/uploads/SGIP_White_Paper_TE_Application_Landscape_Scenarios_12-15-2016_FINAL.pdf>), and any other use cases that support TE. The diagram illustrates the variations in realizations of components of a transactive energy system. The model in this document goes on to produce 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 be able to understand, discuss, evaluate and validate any TE approach. Additionally, the model can be used to look at 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, the grid itself. These can be composed into various devices and systems. For example, a transactive appliance 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.

Obviously, TE is a complex system of systems problem and the detailed simulations will require a diverse set of simulation tools. The model allows a connection to be determined between tools that might be simulating a building management system, or a storage control system and the rest of the TE environment. Allowing specific tools, models, algorithms, and behavior on the balance of the TE environment. The model allows the creating of test of end to end security for each transaction both for control and communications.

Control simulations and algorithms can be layered on the TE model for testing. Control transactions that may be needed to balance the network can be layered on to the TE model to determine reactions and needed latency, or lead times. Pricing and economics can be discussed and tested within the TE model, to determine supporting transactions. With the right economic simulations installed in the model, economic simulation can be tested. With enough customer behavior information, the simulations can simulate within the TE model how customers would react and the trigger prices that might be needed to achieve desired behavior. 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.

To do this, 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 those items that are important to them without having to have an expert help them setup and evaluate their specific aspect of TE.

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.

Obviously, the TE model can be extended by any stakeholder in any specific fashion to deal with simulation of that 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.

The implementation has to faithfully implement a minimum set of specific 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 needs to be able to orchestrate a set of components. These components, like the interfaces can be extended or minimized depending on the question(s) in play. Ideally the components can be simulated by the same experiment controller.

The implementation also needs to support the defined set of grid nodes, resources, controllers, transactive agents, and market simulations to provide the required comparison baseline.

There is also a need to implement a core set of analytics that will be able to evaluate the data from the implemented model against the baseline model and data.

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

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

Section 2: Model -- the description of the core component model itself

Section 3: Scenario -- the orchestration of the model into a "canonical experiment" which enables components implemented and designed to support the abstract interfaces to work in concert in a simulation.

Section 4: Beta Use Case -- a reference use case designed by researchers at PNNL that represent an example instance of the component model

Section 5: Composite and Extended Classes -- some examples of how to merge and/or extend the classes to produce instance models shown in figure 1 above.

## **Model**

The Model consists of a set of abstract components for use in studying transactive energy. Each model 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
* The data types that flesh out the minimum detailed attributes that can be exchanged by the components

### **T****ransactiveEnergyComponents**

These model components expose the key interfaces of the TE Common Model platform. Shown are the roles, their interfaces, and persistent 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).

This section will describe the core components of the model.

At the heart of the model is a simulation grid which represents the energy distribution system. There is a great deal of modeling and experience in describing grids. However, for the purpose of 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. Resources consist of loads and generators (and also storage) devices that can sink or source energy.

There are two types of controllers -- local and supervisory. The local controller is a component that understands the nature of a resource. For example 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 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 in a higher abstraction of supervisory control.

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

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

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

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

This diagram illustrates the classes or roles of components of the TE Challenge Abstract Component Model. 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 (Experiment Manager), and, the core Analytics component that evaluate the data produced during the simulation.

Note that these models provide for a minimum of interoperability and the ability to define an experiment that can exercise the models based on these designs. Any particular 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.



Transactive Energy Components

#### **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.

#### **Weather**

The Weather component provides environmental conditions during a TE experiment. The weather component itself is based on the National Solar Radiation Data Base from NREL know as Typical Model 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/)

| **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 uncert (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 uncert (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 uncert (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 uncert (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 uncert (%) |

|  |  |  |
| --- | --- | --- |
| 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 uncert (%) |

|  |  |  |
| --- | --- | --- |
| 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 uncert (%)  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 uncert (%) |

|  |  |  |
| --- | --- | --- |
| 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 uncert (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 uncert (%) |

|  |  |  |
| --- | --- | --- |
| 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 uncert (%) |

|  |  |  |
| --- | --- | --- |
| 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 uncert (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 uncert (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 uncert (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 uncert (code) |

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

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

|  |  |  |
| --- | --- | --- |
| present\_weather\_uncertainty | double | PresWth uncert (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 uncert (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 uncert (code) |

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

|  |  |  |
| --- | --- | --- |
| wind\_direction\_ncertainty | 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 uncert (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 uncert (%)  1% |

|  |  |  |
| --- | --- | --- |
| m\_SupervisoryController | SupervisoryController |  |

|  |  |  |
| --- | --- | --- |
| m\_Resource | Resource |  |

|  |  |  |
| --- | --- | --- |
| m\_TransactiveAgent | TransactiveAgent |  |

|  |  |  |
| --- | --- | --- |
| m\_Grid | Grid |  |

#### **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.

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| Nodes | Link | List of nodes in Grid |

|  |  |  |
| --- | --- | --- |
| m\_Resource | Resource |  |

#### **LocalController**

A simple base controller that knows how 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. These include logic for such things as voltage regulators or protection devices and determine what state the device should be in and how it might progress to the next state. Additionally, thermostats, loop controllers, etc....

| **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 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. If the downRamp[n].beginRamp attribute is defined for a ramp segment, this is the initial value of the ramp segment; if it is not present the initial value of the ramp equals the ending value of the previous ramp segment. Although the downRamp attribute name implies that the rise is monotonically decreasing, individual array elements may have slopes less than, greater than, or equal to 0. However, the overall trend of the function shall be decreasing. The downRamp function shall measure the time from the load being fully energized until the power is completely depleted. If downRamp is not present, the power decrease to 0 shall be instantaneous. When a curtailable load is partially curtailed (less than the maximumCurtailableDemand) and curtailment is increased, the power will decrease starting at n-th downRamp of the sum from 0 to n of the downRamp[n].rate that is closest to the actualCurtailedDemand. |

|  |  |  |
| --- | --- | --- |
| 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. If the Load.locked attribute is set to TRUE and the: 1) load is not curtailable, the load is immediately locked. The behavior of this operation is a local matter; 2) curtailmentStatus is curtailmentInactive, the load will immediately be locked out from curtailment eligibility; 3) curtailmentStatus is curtailmentNoncompliant, the load will cycle to its curtailmentInactive state then immediately be locked out of curtailment eligibility; 4) If the CurtailableLoad supports multi-stage curtailment, the load will cycle to its curtailmentInactive state for the present curtailment stage and then be locked out of any further curtailment eligibility. 5) curtailmentStatus is curtailmentNoncompliant, the load will cycle to its curtailmentInactive state then immediately be locked out of curtailment eligibility; 6) If the CurtailableLoad supports multi-stage curtailment, the load will cycle to its curtailmentInactive state for the present curtailment stage and then be locked out of any further curtailment eligibility. Loads that are locked will remain in the locked state indefinitely until the Load.locked attribute is reset to FALSE. The mechanism used to unlock the load is a local matter. |

|  |  |  |
| --- | --- | --- |
| 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. If the upRamp[n].beginRamp attribute is defined for a ramp segment, this is the initial value of the ramp segment; if it is not present the initial value of the ramp equals the ending value of the previous ramp segment. Although the upRamp attribute name implies that the rise is monotonically increasing, individual array elements may have slopes less than, greater than, or equal to 0. However, the overall trend of the function shall be increasing. The upRamp function shall measure the time from the load being fully de- energized until the power is completely restored as defined by Load. maximumDemand. If this attribute is not present, the power increase upon restoration shall be instantaneous. When a curtailable load is partially curtailed (less than the maximumCurtailableDemand) and curtailment is reduced, the power will increase starting at n-th recoveryRamp of the sum from 0 to n of the upRamp[n].rate that is closest to the actualCurtailedDemand. |

|  |  |  |
| --- | --- | --- |
| m\_Resource | Resource |  |

#### **Resource**

Grid connected Resources can be loads, generators, 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.

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

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

|  |  |  |
| --- | --- | --- |
| impedance | Impedance | Energy over time step |

|  |  |  |
| --- | --- | --- |
| phases | char |  |

|  |  |  |
| --- | --- | --- |
| power | Power | rate of change of power |

|  |  |  |
| --- | --- | --- |
| status | boolean | status. |

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

#### **SupervisoryController**

Individual agents that control Resources; this does not need to be a one-to-one mapping of controller to 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. Note it is the LocalController that knows the details of the Resource to be controlled. This explicitly does not contain transactive elements, but does contain non- transactive optimizations and operator and consumer actions. NOTE: this element will need to be extremely broad, since there are hundreds of different control variables that one may need access to. 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 building or campus.

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| resources | Resource |  |

|  |  |  |
| --- | --- | --- |
| m\_LocalController | LocalController |  |

#### **BaseModelComponent**

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

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| description | char |  |

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

#### **TransactiveAgent**

Describes a transaction to occur at a given place, including an expression of value, 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.

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| m\_SupervisoryController | SupervisoryController |  |

|  |  |  |
| --- | --- | --- |
| m\_GridController | GridController |  |

|  |  |  |
| --- | --- | --- |
| m\_TransactiveAgent | TransactiveAgent |  |

#### **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.

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| m\_Grid | Grid |  |

|  |  |  |
| --- | --- | --- |
| m\_TransactiveAgent | TransactiveAgent |  |

##### **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. In most cases, the LocalController and Resource will be combined into a single component.

##### **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. In most cases, the LocalController and Resource will be combined into a single component.

#### **Load**

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

#### **Generator**

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

#### **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.

#### **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.

| **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. |

### **Interfaces**

The interfaces for the model represent those messages that are received (subscribed). It is assumed that in order 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 flowed interpreted appropriately to the message mechanism.

#### **Interfaces diagram**

This diagram illustrates the interfaces defined for the TE Challenge Abstract Component Model. Shown are the core components of the model and those interfaces that they realize.

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



Interfaces

#### **iGeneral**

General Interface for all components. Includes Initialization.

#### **iGrid**

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

#### **iLocalControl**

Interface that provides a supervisory control signal to a LocalController.

#### **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.

#### **iResourcePhysicalStatus**

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

#### **iTA**

The TA 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.

#### **iWeatherData**

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

### **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.

Data Descriptions Imported from ASHRAE 201 -- FSGIM. 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

#### **Energy**

A complex vector of energy by phase.

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

#### **GridNode**

This is a node in the grid.

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

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

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

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

#### **AttachNodeDescription**

Parameters to attach a node to the Grid.

| **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

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

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

#### **Current**

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

#### **GridControl**

GridControl service data structure.

#### **GridVoltageState**

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

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

#### **Impedance**

| **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.

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| fromGridNode | GridNode | node 0 id One connecting end of the link object. This will be the name or reference to a node-based object elsewhere in the powerflow model. |

|  |  |  |
| --- | --- | --- |
| id | char | link segment id. |

|  |  |  |
| --- | --- | --- |
| impedance | ComplexNumber | matrix of impedances for each phase |

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

|  |  |  |
| --- | --- | --- |
| name | char |  |

|  |  |  |
| --- | --- | --- |
| 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 | GridNode | node 1 id |

#### **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.

| **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.

| **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.

| **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.

| **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.

| **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. For example a simple 60W incandescent light bulb would likely be modeled by a linear segment (0,0), (100,100). Here as the bulb is energized (dimmed) from off (0%) to fully on (100%), the power needed to energize the device also travels from 0% to 100%. A more complicated device such as a room air-conditioner may have a non-linear power curve. Here, as the coolness setting is adjusted from warmest to coolest, the air conditioner will draw relative more power when set to maximum cooling than when set to minimum cooling. 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.

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

#### **PriceCurveComponent**

A component of a pricing curve.

| **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**

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

#### **ResourceControl**

Parameters used by local controller to control the resource.

#### **ResourceLogicalState**

#### **ResourcePhysicalState**

ResourcePhysicalState describes physical state parameters for the resource.

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

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

#### **ResourceStatus**

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

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

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

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

|  |  |  |
| --- | --- | --- |
| status | boolean | is resource active. |

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

#### **SupervisoryControlSignal**

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

#### **Tender**

| **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 component.

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

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

#### **Transaction**

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

#### **Voltage**

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

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

#### **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 specified in a variety of ways. Below are some identical examples with a simple node object (which is covered in more later in this page).

| **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. |

#### **TimeReference**

A time reference, UTC.

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

#### **GridNodeId**

An integer representing a grid node.

#### **StorageType**

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

| **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.

| **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.

| **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.
3. The interfaces at least provide the minimum 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**

Base scenario for TE experiments. This sequence diagram contains the following:

1. Initialization of all components by the Experiment Manager
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.
* 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.

This diagram 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.

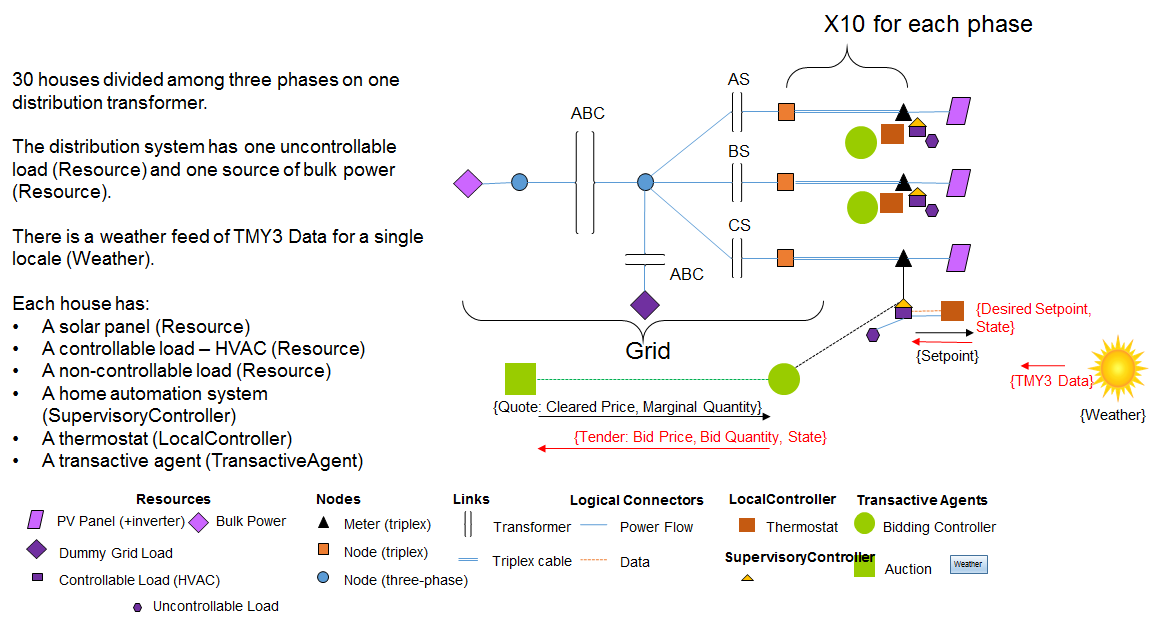
### **TE experiment loop**

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

## **Beta Use Case**

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

The following figure 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 the Abstract Component Model.

### **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.

Once realized on several simulation framework platforms, it provides the basis for comparison and baselining of these platforms.

Shown is a common grid model that supports three sets of 10 house models. One example of each phase-connected house is provided.

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.



Beta Use Case

### **C****ommon Component Inheritance**

This package 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.

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

Shown in the figure is the inheritance model of the Gridlab-D model classes mapped to Abstract Component model.

Note that some parts of the Gridlab-D classes are not distinct but do correspond to the functions of the component model. In these cases, pseudo classes are identified in the diagram 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 identified in this diagram for reference.



Common Component Inheritance

#### **Auction**

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

|  |  |  |
| --- | --- | --- |
| current\_price\_stdev\_24h | double | Current price stdev. |

#### **BillDump**

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

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

|  |  |  |
| --- | --- | --- |
| runtime | char | time. |

#### **Climate**

#### **Clock**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| starttime | char | The start time global variable determine the time at which the simulation begins.ntil 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**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| group | char | Group |

#### **HVAC\_Controller**

gridlab-d "controller"

| **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 |

|  |  |  |
| --- | --- | --- |
| 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: DefF |

|  |  |  |
| --- | --- | --- |
| 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 |  |

|  |  |  |
| --- | --- | --- |
| 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. This will be further discussed later. 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. This will be further discussed later. 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, 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. |

|  |  |  |
| --- | --- | --- |
| m\_HVAC\_Load | HVAC\_Load |  |

#### **HVAC\_Load**

#### **House**

| **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 |  |

|  |  |  |
| --- | --- | --- |
| 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: unit |

|  |  |  |
| --- | --- | --- |
| 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 |  |

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

|  |  |  |
| --- | --- | --- |
| parent | char |  |

|  |  |  |
| --- | --- | --- |
| 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 |  |

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

#### **Inverter**

| **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 |  |

|  |  |  |
| --- | --- | --- |
| 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 | int |  |

|  |  |  |
| --- | --- | --- |
| name | char |  |

|  |  |  |
| --- | --- | --- |
| 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 |  |

|  |  |  |
| --- | --- | --- |
| solar | Solar |  |

|  |  |  |
| --- | --- | --- |
| use\_multipoint\_efficiency | boolean |  |

|  |  |  |
| --- | --- | --- |
| m\_Solar | Solar |  |

#### **Meter**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| bustype | BusType |  |

|  |  |  |
| --- | --- | --- |
| nominal\_voltage | double | Voltage. |

#### **Null\_Controller**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| m\_Unresp\_load | Unresp\_load |  |

#### **Player**

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

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

|  |  |  |
| --- | --- | --- |
| loop | int | number of loops. |

|  |  |  |
| --- | --- | --- |
| name | char |  |

|  |  |  |
| --- | --- | --- |
| value | int | Vlaue. |

#### **Recorder**

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

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

|  |  |  |
| --- | --- | --- |
| parent | char |  |

|  |  |  |
| --- | --- | --- |
| property | char | some properties, such as temperature, humidity,.... |

#### **Solar**

| **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. |

|  |  |  |
| --- | --- | --- |
| 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.

| **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 |  |

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

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

|  |  |  |
| --- | --- | --- |
| name | char |  |

|  |  |  |
| --- | --- | --- |
| outputId | GridNodeId | output Id. |

|  |  |  |
| --- | --- | --- |
| outputs | Phases | outputs |

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

|  |  |  |
| --- | --- | --- |
| 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 powerflow model. (Unit:N/A) |

#### **Transformer\_configuration**

| **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 |  |

|  |  |  |
| --- | --- | --- |
| 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**

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

|  |  |  |
| --- | --- | --- |
| groupId | char | gourp Id. |

|  |  |  |
| --- | --- | --- |
| m\_Triplex\_line\_configuration | Triplex\_line\_configuration |  |

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

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

|  |  |  |
| --- | --- | --- |
| name | char |  |

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

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

| **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 |  |

|  |  |  |
| --- | --- | --- |
| m\_Triplex\_line\_conductor | Triplex\_line\_conductor | m triplex line conductor. |

#### **Triplex\_meter**

| **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. 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 |  |

|  |  |  |
| --- | --- | --- |
| 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 |  |

|  |  |  |
| --- | --- | --- |
| 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**

**Triplex Load** Triplex load is similar to load and ZIPload in that load can be specified as a base load, then a ZIP fraction applied to that base load. The load can be place on phase 1 (120V), phase 2 (120V) or phase 12 (240V).

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

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

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

|  |  |  |
| --- | --- | --- |
| current\_fraction\_A | double | tTis 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 A 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 |  |

|  |  |  |
| --- | --- | --- |
| 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**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| base\_power | double | In similar format as ZIPload 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 | In similar format as ZIPload 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**

| **Name** | **Type** | **Notes** |
| --- | --- | --- |
| m\_ZIPLoad | ZIPLoad | ZIPLoad. |

### 

### **Grid**

This package describes the beta use case grid model.

#### **Grid diagram**

This package describes the beta use case grid model.



Grid

#### **F1\_Transformer1**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

#### **Link01**

#### **Link03**

#### **Link04**

#### **Link05**

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

#### **PHASE\_A**

#### **PHASE\_B**

#### **PHASE\_C**

#### **TLCFG**

#### **bulkPower**

#### **substation\_root**

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

Substation object should be replaced by this meter.

bustype attributes is missed here.

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

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

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

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

#### **unresp\_load**

### 

### **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**



PhaseAHouse

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

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

#### **dryer\_A5**

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

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

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

#### **lights\_A5**

#### **microwave\_A5**

#### **range\_A5**

#### **refrigerator\_A5**

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

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

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

### 

### **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**



PhaseBHouse

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

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

#### **dryer\_B0**

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

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

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

#### **lights\_B0**

#### **microwave\_B0**

#### **range\_B0**

#### **refrigerator\_B0**

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

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

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

### 

### **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**



PhaseCHouse

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

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

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

#### **dishwasher\_C1**

#### **dryer\_C1**

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

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

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

#### **lights\_C1**

#### **microwave\_C1**

#### **range\_C1**

#### **refrigerator\_C1**

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

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

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

### 

### **Enumerations**

Enumerations used in the beta use case model.

#### **Enumerations diagram**



Enumerations

#### **BillMode**

| **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**

| **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.

| **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

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

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

#### **CoolingSystemType**

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

| **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)

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

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

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

#### **GeneratorMode**

| **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**

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

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

#### **HeatingSystemType**

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

| **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.

| **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**

| **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)

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

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

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

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

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

#### **MotorModel**

| **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.

| **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

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

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

#### **ResolveMode**

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

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

#### **SystemMode**

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

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

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

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

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

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

### **auction**

### **climate**

## **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.

### **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.

### **ResourceWithLocalController**

Contains Resource with Local Controller.

### **CustomInterface**

A custom extension interface for one of the core components.