NIST Special Publication 1900-750

The Transactive Energy Abstract Component Model

Martin Burns
Eugene Song
David Holmberg

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.1900-750

CYBER-PHYSICAL SYSTEMS

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Special Publication 1900-750

The Transactive Energy Abstract Component Model

Martin Burns
Eugene Song
David Holmberg
Smart Grid and Cyber-Physical Systems Program Office
Engineering Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.1900-750

November 2018

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Under Secretary of Commerce for Standards and Technology

Publications in the SP 1900 subseries present information of interest to the cyber-physical systems (CPS) community, where CPS are defined as smart systems that include engineered interacting networks of physical and computational components. The series was established in 2016 by the Smart Grid and Cyber-Physical Systems Program Office of the NIST Engineering Laboratory to provide a separate identity for CPS and Internet of Things publications, including those concerned with the foundations of CPS, CPS testbed science, and CPS applications, e.g., smart grid, smart cities and intelligent transportation. The series reports on research, guidelines, and outreach efforts in CPS, and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 1900-750
Natl. Inst. Stand. Technol. Spec. Publ. 1900-750, 54 pages (November 2018)
CODEN: NSPUE2

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.1900-750

Revision Tracking

Version	Date	Editor	Changes
1.0	November 2018	Martin Burns	First Release

Table of Contents

Table of Figures

Table of Tables

Acknowledgement

NIST would like to acknowledge the valuable leadership, collaboration and contributions of the following individuals who helped guide the participants through the modeling activity:

Jason Fuller, Pacific Northwest National Laboratory

Maria Ilic, Massachusetts Institute of Technology/ Carnegie Mellon University

Chris Irwin, Department of Energy

Himanshu Neema, Vanderbilt University

Executive Summary

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

The tiger team created the model to provide a basis for common discussions, like the Institute of Electrical and Electronics Engineers (IEEE) standard feeder models that the industry has used for more than a decade to test changes to the electric power grid through simulations. The feeder models have allowed side-by-side comparisons, speeding up development of better simulation tools, more meaningful predictions of power system performance, and understanding of modeling and simulation in many university power engineering programs.

NIST is assembling software and/or hardware implementations that realize the interfaces of this model within its Universal CPS Environment for Federation (UCEF) built by NIST and Vanderbilt [2]. This serves as the foundation for NIST's Transactive Energy Analytical Measurement System (TEAMS). TEAMS is being built on NIST's advanced Cyber-Physical Systems Testbed technology to fill today's gap in the ability to measure performance across the many different TE models and implementations being considered by utilities across the nation. A similar platform-based on PNNL technology is now available [3].

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

1. Overview

1.1. Introduction

The evolving smart grid, with increased use of renewable energy generation and distributed energy management technologies, offers the potential for significant efficiency improvements through market-based transactive exchanges between energy producers and energy consumers. To understand this potential and support technology developers and policy makers, the smart grid community will require simulation tools and platforms that can be used to explore the benefits and impacts of alternative ways to create and operate energy systems. The National Institute of Standards and Technology (NIST) is charged by the 2007 Energy Independence and Security Act (EISA) [4] with facilitation of interoperability standards to enable successful implementation of the evolving cyber-physical national electric grid system known as the smart grid. As part of this effort, NIST has produced a framework [5] to describe its understanding of the state of the smart grid standards. The use of economic signals to mitigate demands was identified as key tool in grid evolution.

In this regard, transactive energy (TE) has been identified as an important area of research for the development of a modern electric grid with important technical challenges as well as needed interoperable approaches. The U.S. Department of Energy (DOE) GridWise Architecture Council has published a Transactive Energy Framework [6] that defines TE broadly as, "a system of economic and control mechanisms that allows the dynamic balance of supply and demand across the entire electrical infrastructure using value as a key operational parameter." To achieve this goal the Transactive Energy Modeling and Simulation Challenge for the Smart Grid (TE Challenge) [7] brought researchers and companies with simulation tools together with other grid stakeholders to demonstrate modeling and simulation platforms while applying TE approaches to real grid problems.

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

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

The Transactive Energy Abstract Component Model is this foundational model that aids understanding and communications of TE co-simulation. The model was the product of a tiger team effort engaged by the National Institute of Standards and Technology (NIST), Pacific Northwest National Lab (PNNL), Vanderbilt University, and Carnegie Mellon University during the summer of 2016 in support of the NIST Transactive Energy Challenge [1].

The tiger team created this model to provide a basis for common discussions, like the IEEE standard feeder models that the industry has used for more than a decade to test changes to

simulations. The feeder models have allowed side-by-side comparisons, speeding up development of better simulation tools, and understanding of modeling and simulation in many university power engineering programs. The TE abstract component model should be able to provide similar support for TE in the industry. However, while the IEEE feeder models provide standard sets of physical components attached to a physical grid, the model goes beyond this to describe abstract components with defined interfaces to allow connecting these components in a TE simulation that includes the consideration of loads, generators, controllers and markets in addition to the grid itself.

The intent 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. Modeling of power systems inside the grid, although important to the realization of grid simulations was outside the scope of the abstract component modeling activity. It was assumed that existing modeling of the grid component, for example using GridLAB-D [8], were sufficient for the model.

A notional diagram of the model is given in Figure 1.

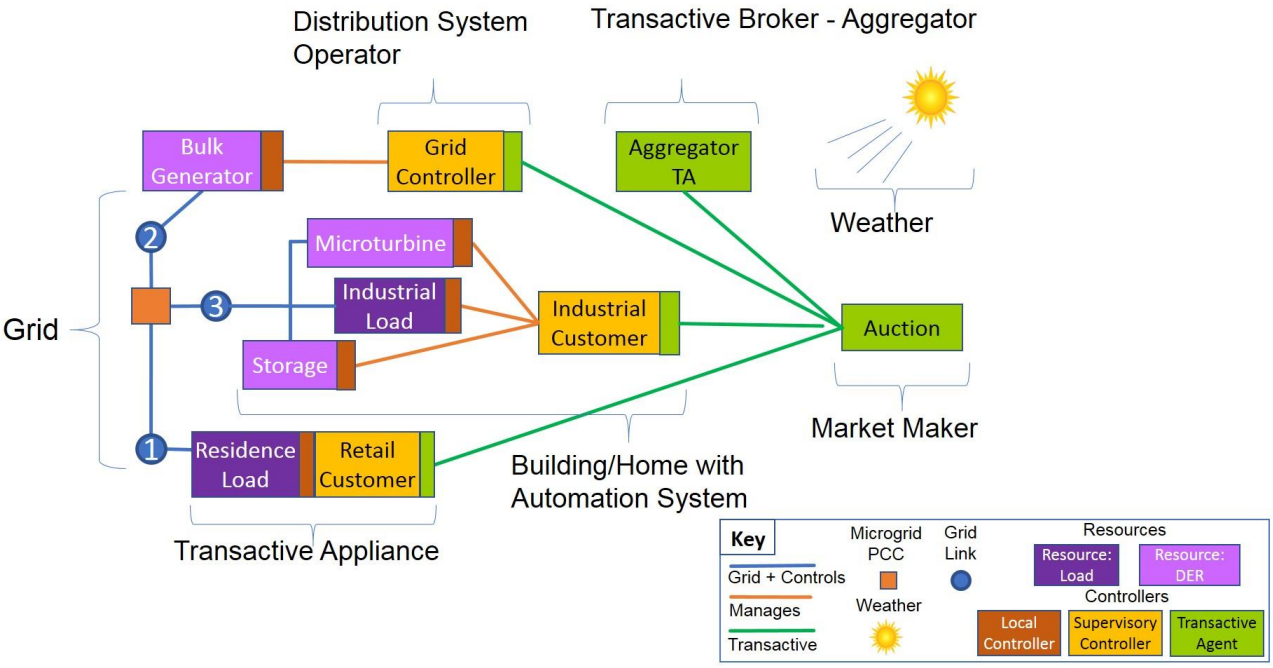

Figure 1: Notional Topology Illustrating TE Model Building Blocks

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

While Figure 1 shows a system that is much simpler than a real-world situation, it is complete from a testing point of view. There is at least one actor of each major type and one component of each major type. The model can be used to evaluate any of the Smart Grid Interoperability Panel

(SGIP) TE Landscape Scenarios [9], and any other use cases that support TE. The diagram illustrates the variations in realizations of components of a transactive energy system. This document goes on to describe a set of model components that can be assembled and otherwise extended to simulate transactive systems of arbitrary design.

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

1.2. TE Model Development and Use

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

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

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

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

The overall model allows stakeholders to not only test sample implementations, 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 model so that any stakeholder has a starting toolbox from which to tinker. Then, stakeholders can create their own improved tools for the specific areas they are interested in, while running the common

tools for other aspects. This shortens development times for stakeholders and greatly lowers the barriers to entry for involvement.

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

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

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

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

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

1.3. Structure of This Document

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

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

2. Model

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

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

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

In the model, names are formatted in camel case where each word is begun with a capital letter. By this means the phrase "by this means" becomes "ByThisMeans". Major names use “UpperCamelCase” where the first letter is capitalized and minor names, such as attributes of classes us “lowerCamelCase” where the first letter is lower case.

In GridLAB-D discussed in the Beta Use Case section below, underscore-delimited compound naming is used. By this means becomes "by_this_means" and where all characters are lower case.

2.1. Transactive Energy Components

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

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

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

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

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

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

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

2.1.1. Transactive Energy Components diagram

Figure 2 illustrates the classes or roles of components of the 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 (ExperimentManager), and, the core analytics component (Analytics) that evaluate the data produced during the simulation.

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

The components in Fig. 2 can be seen to have "lollipop" symbols — a circle and a line — sticking out of the left side of the component symbol. These represent the interfaces realized by the component. See Section 2.2 for the interface definitions.

Some of the components have state information necessary for the simulation. Others are defined only through the interfaces of data exchanged. Naturally, implementations of all the components will require some state information, but this is not required to be standardized in the model. Therefore, some of the components documented in this section will include the state detail and others will not.

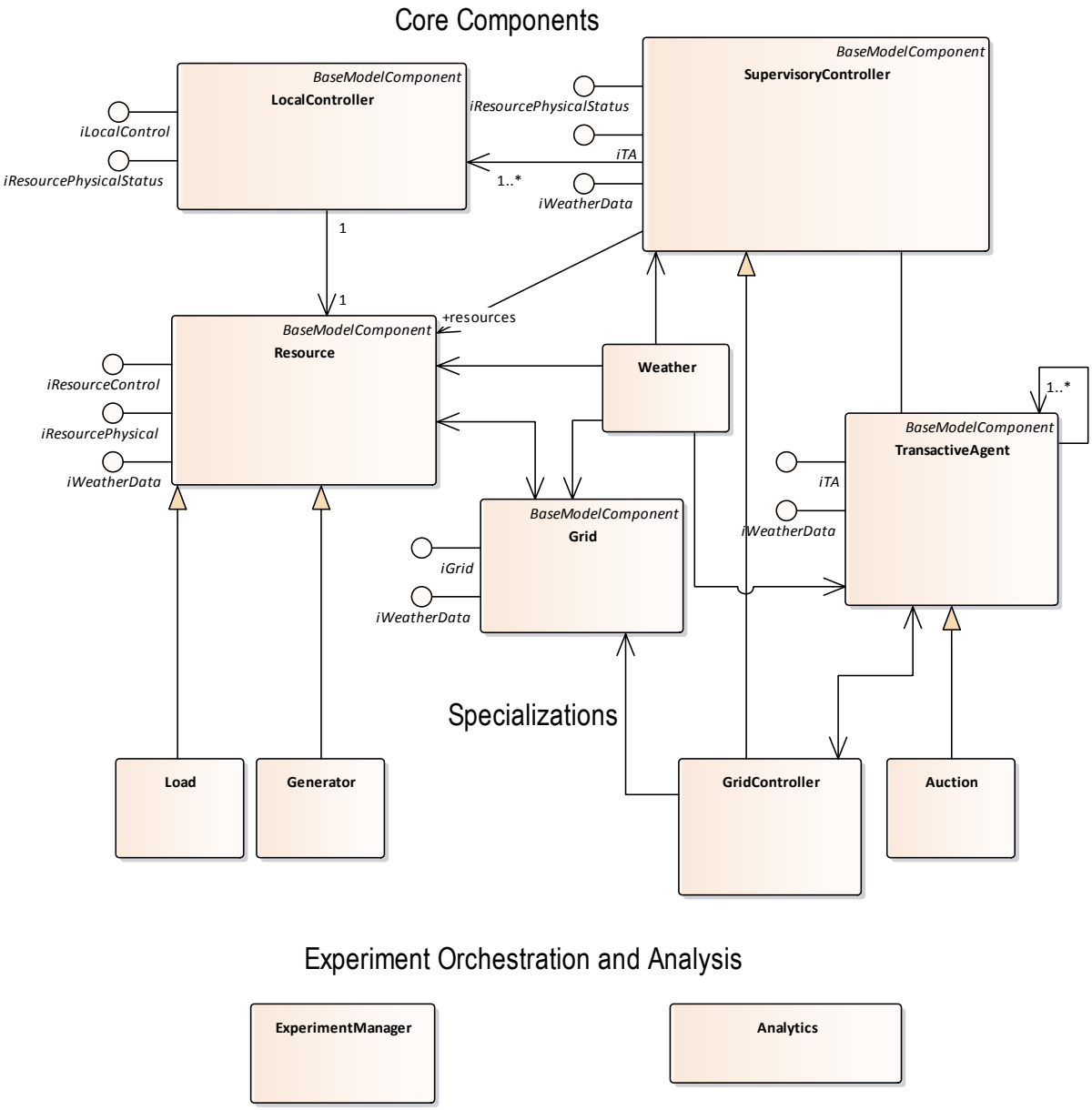

Figure 2: Transactive Energy Components

2.1.2. LocalController

A simple base controller that contains logic to control a resource based on a dimensionless modulation setting received from a SupervisoryController. It does not have awareness of any other component of the system. Local Controllers include not only load and generator controllers, but also controllers of voltage regulators and protection devices and determine what state the device should be in and how it might progress to the next state. Example controllers include thermostats and loop controllers. Local Controller component attributes are provided in Table 1.

Table 1: LocalController Detail

Name	Type	Notes
actualDemand	float	This attribute defines the power being consumed by the device (as measured by the present subinterval demand) at the present time.
demandLimits	PowerRatings	The operational demand characteristics and their associated end points for the load.
downRamp	PowerRampSegmentType	This attribute defines the reduction in power over time when a load or generator being partially or fully de-energized has a complex load reduction profile. For each element of the Load.downRamp array, the downRamp[n].rate defines the amount of power decrease and the downRamp[n].duration defines the length of time in seconds upon which the decrease is in effect.
locked	Boolean	This attribute defines whether the load is locked and therefore ineligible for curtailment; or unlocked and available for curtailment. Load locking behavior changes depending on the load's curtailmentStatus attribute value at the time the load was locked.
status	LoadStatusType	This attribute defines the current status of the load. For non-curtailable loads, it provides the present communication status and reliability of the data. For curtailable loads, it also defines if the load is eligible for curtailment or why it is ineligible for curtailment.
upRamp	PowerRampSegmentType	This attribute defines the increase in power over time when a load being partially or fully energized has a complex demand restoration profile. For each element of the Load.upRamp array, the upRamp[n].rate defines the amount of power increase and the upRamp[n].duration defines the length of time in seconds upon which the increase is in effect.

2.1.3. SupervisoryController

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

This explicitly does not contain transactive elements but does contain non-transactive optimizations and operator and consumer actions. The implementation of this component may be extremely broad, since there are many different control variables that may drive a supervisory control algorithm. For example, a building control system which has setpoints and schedules, and other customer preferences may act as the SupervisoryController for all Resources in a home, building or campus. Supervisory Controller component attributes are provided in Table 2.

Table 2: SupervisoryController Detail

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

2.1.4. Resource

Grid connected Resources can be loads, generators, or storage devices. They consume or generate energy. Resources are 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, Table 3. The resource details given here 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.

Table 3: Resource Detail

Name	Type	Notes
current	Current	Current consumed (positive) by Resource by phase.
gridNodeId	char	Identifies grid node load is connected to.
impedance	Impedance	Resource complex impedance by phase and across phases.
phases	Phases	Electrical phases.
power	Power	Power consumed (positive) by Resource by phase.
status	boolean	Status of resource — active (true) or inactive (false).
voltage	Voltage	Voltage by phase.

2.1.5. Weather

The Weather component provides environmental conditions during a TE Challenge experiment. The properties of the weather component, Table 4, are based on the National Solar Radiation Data Base from NREL know as Typical Model Year (TMY3) [11].

Table 4: Weather Detail

Name	Type	Notes
aerosol_optical_depth	double	The broadband aerosol optical depth per unit of air mass due to extinction by the aerosol component of the atmosphere. Unit: AOD [unitless] Resolution: 0.001
aerosol_optical_depth_source	char	AOD source.

Name	Type	Notes
aerosol_optical_depth_uncertainty	double	AOD uncertainty (code).
albedo	double	The ratio of reflected solar irradiance to global horizontal irradiance. Unit: Alb [unitless] Resolution: 0.01
albedo_source	char	Alb source.
albedo_uncertainty	double	Alb uncertainty (code).
latitude	double	Site latitude. Unit: Decimal degree
ceiling_height	double	Height of the cloud base above local terrain (77777 = unlimited). Unit: CeilHgt (m) Resolution: 1m
ceiling_height_source	char	CeilHgt source.
ceiling_height_uncertainty	double	CeilHgt uncertainty (code).
date	char	Date of data record. Unit: mm/dd/yyyy
dew_point_temperature	double	Dew-point temperature at the time indicated. Unit: Dew-point (Degree C) Resolution: 0.1degree
dew_point_temperature_source	char	Dew-point source.
dew_point_temperature_uncertainty	double	Dew-point uncertainty (code).
diffuse_horizontal_illuminance	double	Average amount of illuminance received from the sky (excluding the solar disk) on a horizontal surface during the 60-minute period ending at the timestamp. Unit: DH illum (lux) Resolution: 100 Ix
diffuse_horizontal_illuminance_source	char	DH illum source.
diffuse_horizontal_illuminance_uncertainty	double	DH illum uncertainty (%).
diffuse_horizontal_irradiance	double	Amount of solar radiation (modeled) received in a collimated beam on a surface normal to the sun during the 60-minute period ending at the timestamp. Unit: DHI (W/m^2)

Name	Type	Notes
		Resolution: 1Wh/m^2
diffuse_horizontal_irradiance_source	char	DHI source.
diffuse_horizontal_irradiance_uncertainty	double	DHI uncertainty (%).
direct_normal_illuminance	double	Average amount of direct normal illuminance received within a 5.7° field of view centered on the sun during 60-minute period ending at the timestamp. Unit: DN illum (lux) Resolution: 100 IX
direct_normal_illuminance_source	char	DN illum source.
direct_normal_illuminance_uncertainty	double	Uncertainty based on random and bias error estimates. Unit: DN illum uncertainty (%) Resolution: 1%
direct_normal_irradiance	double	Amount of solar radiation (modeled) received in a collimated beam on a surface normal to the sun during the 60-minute period ending at the timestamp. Unit: DNI (W/m^2) Resolution: 1 Wh/m^2
direct_normal_irradiance_source	char	DNI source.
direct_normal_irradiance_uncertainty	double	DNI uncertainty (%).
dry_bulb_temperature	double	Dry-bulb temperature at the time indicated. Unit: Dry-bulb (Degree C) Resolution: 0.1 degree
dry_bulb_temperature_source	char	Dry-bulb source.
dry_bulb_temperature_uncertainty	double	Dry-bulb uncertainty (code).
elevation	double	Site elevation. Unit: Meter
extra_terrestrial_radiation	double	Amount of solar radiation received on a horizontal surface at the top of the atmosphere during the 60-minute period ending at the timestamp. Unit: W/m^2 Resolution: 1Wh/m^2

Name	Type	Notes
extra_terrestrial_radiation_normal	double	Amount of solar radiation received on a surface normal to the sun at the top of the atmosphere during the 60-minute period ending at the timestamp. Unit: W/m^2 Resolution: 1 Wh/m^2
global_horizontal_illuminance	double	Average total amount of direct and diffuse illuminance received on a horizontal surface during the 60-minute period ending at the timestamp. Unit: GH illum (lux) Resolution: 100 Ix
global_horizontal_illuminance_source	char	GH illum source.
global_horizontal_illuminance_uncertainty	double	Global illum uncertainty (%).
global_horizontal_irradiance	double	Total amount of direct and diffuse solar radiation received on a horizontal surface during the 60-minute period ending at the timestamp. Unit: W/m^2 Resolution: $1Wh/m^2$
global_horizontal_irradiance_source	char	GHI source.
global_horizontal_irradiance_uncertainty	double	GHI uncertainty (%).
horizontal_visibility	double	Distance to discernable remote objects at the time indicated (7777 = unlimited). Unit: Hvis (m) Resolution: 1m
horizontal_visibility_source	char	Hvis source.
horizontal_visibility_uncertainty	double	Hvis uncertainty (code).
liquid_precipitation_depth	double	The amount of liquid precipitation observed at the indicated time for the period indicated in the liquid precipitation quantity field. Unit: Lprecip depth (mm). Resolution: 1mm
liquid_precipitation_depth_source	char	Lprecip source.
liquid_precipitation_depth_uncertainty	double	Lprecip uncertainty (code).
liquid_precipitation_quantity	double	The period of accumulation for the liquid precipitation depth field.

Name	Type	Notes
		Unit: Lprecip quantity (hr) Resolution: 1hr
longitude	double	station longitude. Unit: Decimal Degree
opaque_sky_cover	double	Amount of sky dome covered by clouds or obscuring phenomena that prevent observing the sky or higher cloud layers at the time indicated. Unit: OpqCld (tenths) Resolution: 1 tenth
opaque_sky_cover_source	char	OpqCld source.
opaque_sky_cover_uncertainty	double	OpqCld uncertainty (code).
precipitable_water	double	The total precipitable water contained in a column of unit cross section extending from the earth's surface to the top of the atmosphere. Unit: Pwat (cm) Resolution: 0.1cm
precipitable_water_source	char	Pwat source.
precipitable_water_uncertainty	double	Pwat uncertainty (code).
present_weather	double	PresWth (METAR code).
present_weather_source	char	PresWth source.
present_weather_uncertainty	double	PresWth uncertainty (code).
pressure	double	Station pressure at the time indicated. Unit: Pressure (mbar) Resolution: 1mbar
pressure_source	char	Pressure source.
pressure_uncertainty	double	Pressure uncert (code).
relative_humidity	double	Relative humidity at the time indicated. Unit: RHum (%) Resolution: 1%
relative_humidity_source	char	RHum source.
relative_humidity_uncertainty	double	RHum uncertainty (code).
station_id_code	int	station identifier code.

Name	Type	Notes
station_name	char	Station name.
station_state	char	Station state.
time	char	hh:mm:ss local time.
time_zone	double	Station time zone. Hours from Greenwich, negative west.
total_sky_cover	double	Amount of sky dome covered by clouds or obscuring phenomena at the time indicated. Unit: TotCld (tenths) Resolution: 1 tenth
total_sky_cover_source	char	TotCld source.
total_sky_cover_uncertainty	double	TotCld uncertainty (code).
wind_direction	double	Wind direction at the time indicated. Unit: Wdir (degrees) Resolution: 10 degree
wind_direction_uncertainty	double	Wdir uncert (code).
wind_direction_source	char	Wdir source.
wind_speed	double	Wind speed at the time indicated. Unit: Wspd (m/s) Resolution: 0.1 m/s
wind_speed_source	char	Wspd source.
wind_speed_uncertainty	double	Wspd uncertainty (code).
zenith_luminance	double	Average amount of luminance at the sky's zenith during the 60-minute period ending at the timestamp. Unit: Zenith lum (cd/m^2) Resolution: 10 cd/m^2
zenith_luminance_source	char	Zenith lum source.
zenith_luminance_uncertainty	double	Uncertainty based on random and bias error estimates. Unit: Zenith lum uncertainty (%) 1%

2.1.6. 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, Table 5. Resources such as loads and generators are "attached" to the nodes of the Grid.

Table 5: Grid Detail

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

2.1.7. TransactiveAgent

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

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

2.1.8. Load

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

2.1.9. Generator

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

2.1.10. 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.

2.1.11. 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.

2.1.12. 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.

2.1.13. Analytics

The Analytics component analyzes data and produces metrics of the scenario. Each instance of Analytics is generated for a step in the time sequence of the experiment based on the timing of messaging for each phase of the message loop. Analytics component attributes are shown in Table 6.

Table 6: Analytics Detail

Name	Type	Notes
aggregatedLoadsByHousehold	Energy	Aggregated load by household.
generationProfile	Energy	Generation by generator.
gridPower	Power	Power provided by the Grid.
loadProfile	Energy	Energy consumed by each load.
priceNegotiations	Tender	Sequence of all tenders.
realizeMarketPricing	Quote	Realized Market price quotes.
Voltage	Voltage	Voltage at every link point in pairs in link order and fromVoltage prior to toVoltage.

2.1.14. BaseModelComponent

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

Table 7: BaseModelComponent Detail

Name	Type	Notes
Description	char	General Transactive Energy Model Component provides for the initialization of simulation model components.
Name	char	Name of the component

2.2. Interfaces

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

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

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

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

2.2.1. Interfaces diagram

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

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

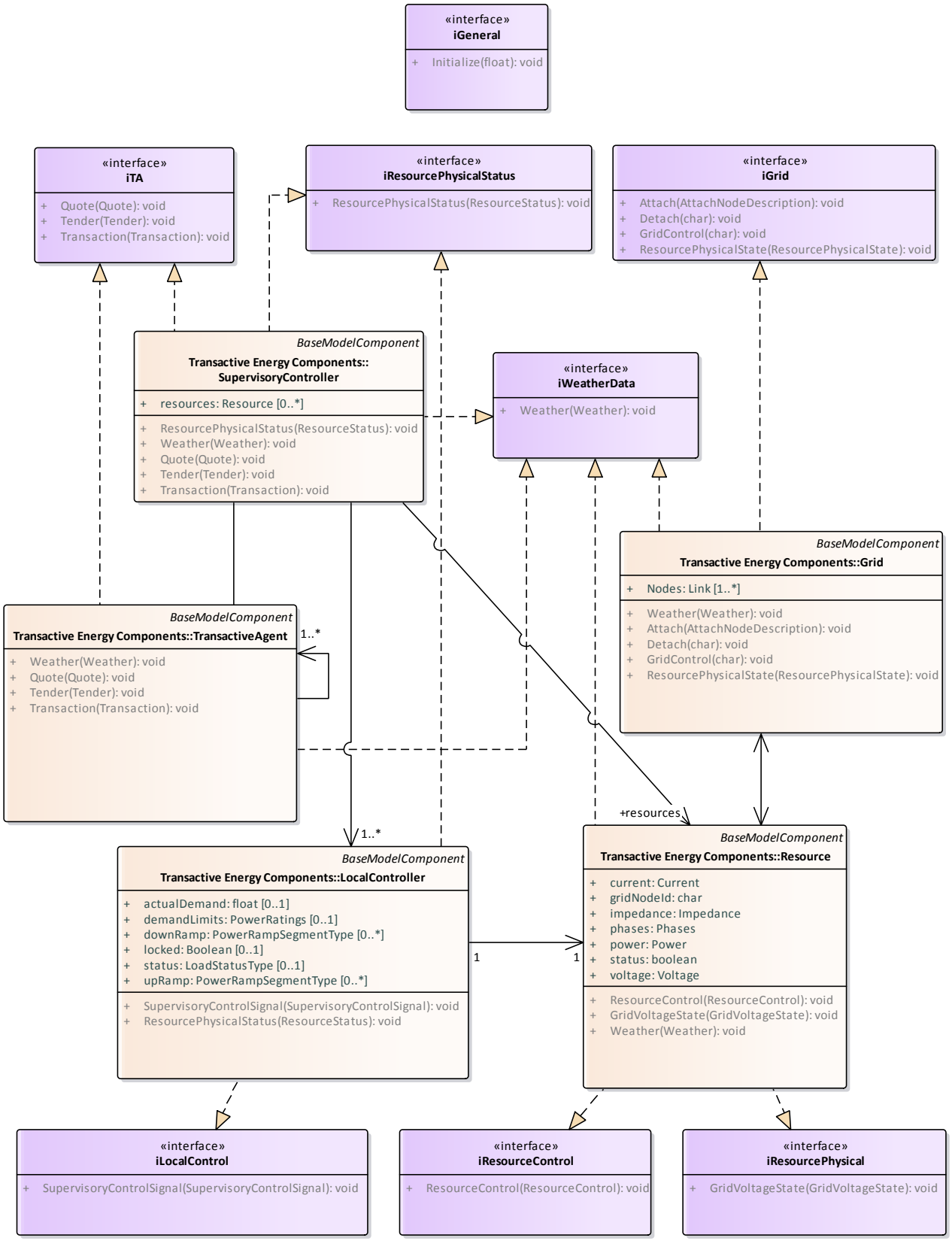

Figure 3: Interfaces

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

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

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

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

2.2.6. iLocalControl
This interface provides a supervisory control signal to a LocalController.

2.2.7. 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 is a very large range of potential data that must be exchanged over this interface for any particular kind of resource and this will not be standardized in this specification to any degree. Note: in many cases, the LocalController and Resource will be combined into a single device — e.g. a hot water heater (load resource) comes assembled including its thermostat (local controller).

2.2.8. 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.

2.2.9. iGeneral
General Interface for all components including Initialization.

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

2.3.1. DataTypes diagram
The diagram in Fig. 4 presents data types in three categories:

- Component Model Data Structures — defines the main data structures that represent grid state;
- Interface Parameter Classes — defines the content of interface messages; and
- Data descriptions imported from ASHRAE 201 — Facility Smart Grid Information Model (FSGIM) [12]. 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.

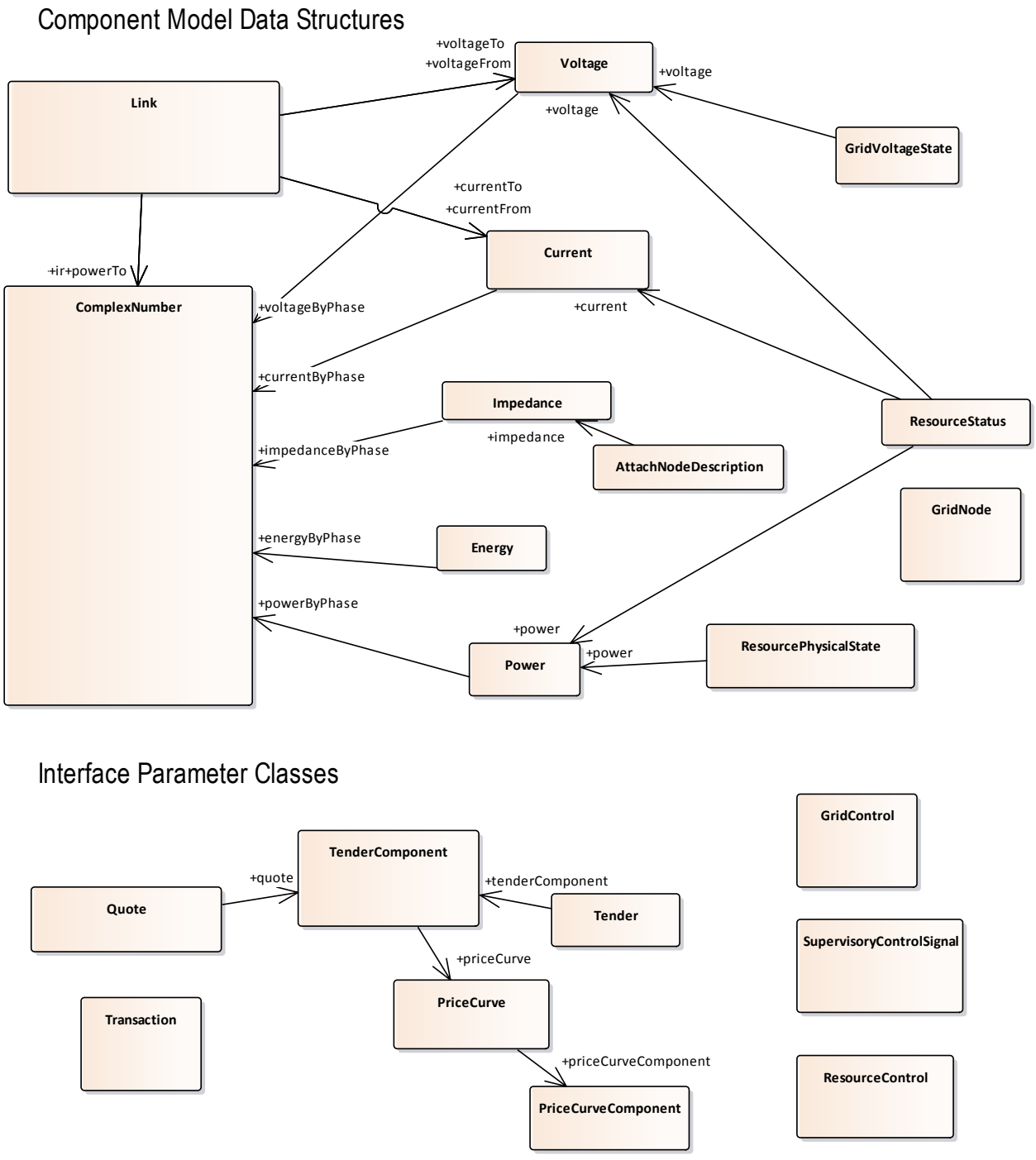

Figure 4: Data Types

2.3.2. Data Enumerations diagram

The diagram in Fig. 5 presents enumerations and primitive data types that are part of the model.

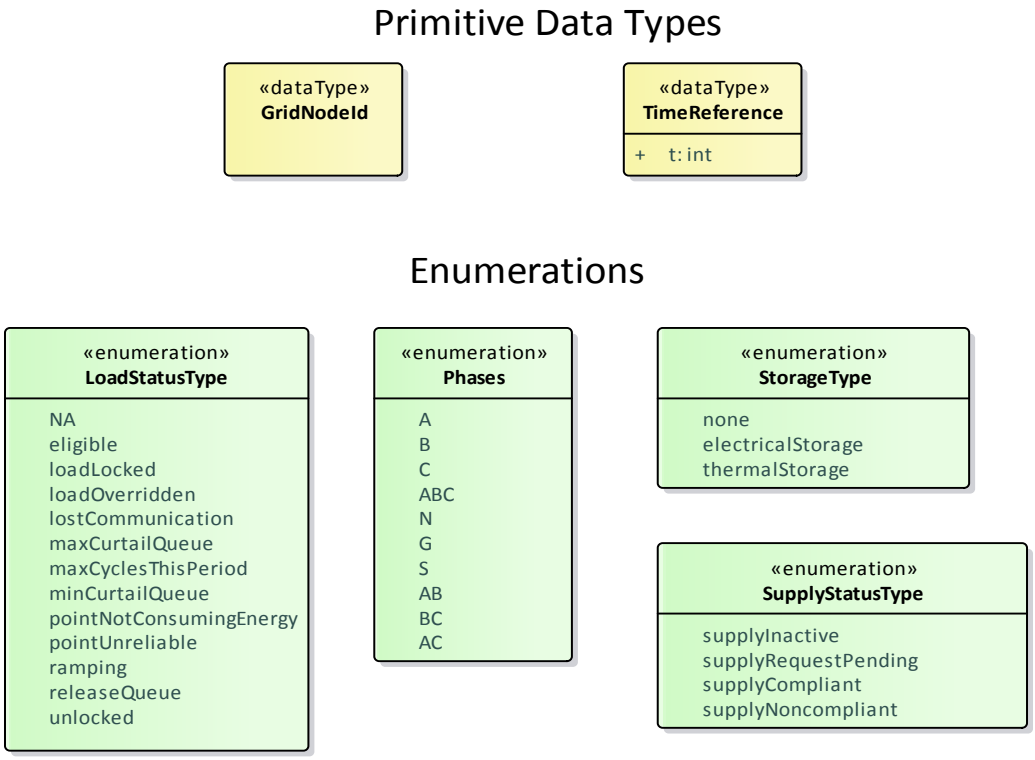

Figure 5: Data Enumerations

2.3.3. Energy

A complex vector of energy by phase, Table 8.

Table 8: Energy Detail

Name	Type	Notes
energyByPhase	ComplexNumber	Energy by phase.

2.3.4. GridNode

This is a node in the grid with details in Table 9.

Table 9: GridNode Detail

Name	Type	Notes
current	Current	Current flowing into the node.
id	GridNodeId	This is id of grid node.
power	Power	Power into the node.
voltage	Voltage	Voltages at node.

2.3.5. AttachNodeDescription

Parameters to attach a node to the Grid, Table 10.

Table 10: AttachNodeDescription Detail

Name	Type	Notes
gridNodeID	char	Node identifier to attach to in the Grid model.
impedance	Impedance	Impedance matrix for phase connections.
phases	Phases	Phases for attachment.

2.3.6. ComplexNumber

A complex number, Table 11.

Table 11: ComplexNumber Detail

Name	Type	Notes
imaginary	float	The imaginary part.
real	float	The real part.

2.3.7. Current

A complex set of currents by phase, in Amperes, Table 12.

Table 12: Current Detail

Name	Type	Notes
currentByPhase	ComplexNumber	Current by phase.

2.3.8. GridControl

GridControl service data structure. Not defined at this time.

2.3.9. GridVoltageState

Details of voltage by phase at a node in the grid in Volts, Table 13.

Table 13: GridVoltageState Detail

Name	Type	Notes
phases	Phases	Phase.
voltage	Voltage	Voltage.

2.3.10. Impedance

A set of complex impedances by phase and across phases, Table 14.

Table 14: Impedance Detail

Name	Type	Notes
impedanceByPhase	ComplexNumber	Impedance by phase.

2.3.11. Link

A Link, Table 15, 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.

Table 15: Link Detail

Name	Type	Notes
fromGridNode	GridNodeId	One connecting end of the link object. This will be the reference to a node-based object elsewhere in the powerflow model.
id	char	Link segment id.
impedance	Impedance	Matrix of impedances for each phase.
length	float	Length of link segment in meters.
name	char	Link name.
phases	Phases	Sequence of phases from A, B, C, N, L1, L2. e.g. three phase 4 wire — ABCN Note phase order indicates the index in to the vector array of impedances or power or voltage. Voltage relative to N.
status	boolean	True if connectivity established false if connectivity denied.
toGridNode	GridNodeId	Identifies the second node of the link.

2.3.12. PiecewiseLinearSegment

The PiecewiseLinearSegment class, Table 16, 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.

Table 16: PiecewiseLinearSegment Detail

Name	Type	Notes
desiredFractionOfFullRatedOutputBegin	float	This attribute defines the starting x-coordinate of the straight line segment.
desiredFractionOfFullRatedOutputEnd	float	This attribute defines the ending x-coordinate of the straight line segment.
requiredFractionOfFullRatedInputPowerDrawnBegin	float	This attribute defines the starting y-coordinate of the straight line segment.
requiredFractionOfFullRatedInputPowerDrawnEnd	float	This attribute defines the ending y-coordinate of the straight line segment.

2.3.13. Power

A complex vector of power by phase, Table 17.

Table 17: Power Detail

Name	Type	Notes
powerByPhase	ComplexNumber	

2.3.14. PowerCurve

The PowerCurve class, Table 18, 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.

Table 18: PowerCurve Detail

Name	Type	Notes
maximumReactivePower	float	This attribute defines the maximum reactive power consumed (or supplied) by the device in units specified in PowerReactiveType.
maximumRealPower	float	This attribute defines the maximum real power consumed (or supplied) by the device in units specified in PowerRealType.
reactivelPowerCurve	PiecewiseLinearSegment	This attribute defines the reactive component of a single piecewise linear curve mapping the percentage of power consumed by the device as a function of the present level of operation of the device.
realPowerCurve	PiecewiseLinearSegment	This attribute defines the real component of a single piecewise linear curve mapping the percentage of power consumed by the device as a function of the present level of operation of the device.

2.3.15. PowerRampSegmentType

The PowerRampSegmentType data structure, Table 19, 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.

Table 19: PowerRampSegmentType Detail

Name	Type	Notes
beginRamp	float	The attribute defines the quantity of power at the start of the ramp segment. If this attribute is not defined in the segment, the start of the ramp is assumed to be the end of the ramp of the previous segment.
duration	int	The attribute defines the time horizon in seconds upon which the associated rise or drop is valid.
rampToCompletion	boolean	The attribute defines whether the ramping up or down of this load may be halted in midstream (false) or once started must complete through all segments of the ramp (true). As an example, a multistate fan may only use a portion of the ramp, as it sequences from low to medium to high speed levels (false); whereas, a production line, once started, may need to run through its complete set of ramp segments (true). If the attribute is not defined it is assumed to be false.
rate	float	The attribute defines rate of rise (positive value) in demand or the rate of drop (negative value) in demand when a load either powers up or shuts down respectively. Its sister attribute, duration, defines the time frame upon which the rate is defined.

2.3.16. PowerRatings

The PowerRatings class, Table 20, 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.

Table 20: PowerRatings Detail

Name	Type	Notes
activePowerCurve	int	This attribute defines the index into the zero based array of powerCurves indicating which powerCurve is presently active.
adjustedFullDRPower	float	This attribute defines the minimum expected power draw of a load (or the maximum power supplied by a generator) during operation. This value differs from the rated power since it may take into account operational considerations such as environmental, equipment safety or regulatory conditions.
adjustedNoDRPower	float	This attribute defines the maximum expected power draw of a load (or the minimum power supplied by a generator) during operation. This value differs from the rated power since it may take into

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

2.3.17. PriceCurve

A price curve, Table 21, depends on sign of the PriceCurveComponent.quantity, it can be price of supply (+) or demand (-).

Table 21: PriceCurve Detail

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

2.3.18. PriceCurveComponent

A component of a pricing curve, Table 22.

Table 22: PriceCurveComponent Detail

Name	Type	Notes
price	float	Price of commodity.
quantity	float	Quantity of commodity (signed number) can be supply (+) or demand (-).
type	char	Type of commodity — W, Var, V, Frequency, Wh, ...

2.3.19. Quote

A quote exchanged by the TransactiveAgents, Table 23.

Table 23: Quote Detail

Name	Type	Notes
quote	TenderComponent	Array of tender components.

2.3.20. ResourceControl

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

2.3.21. ResourcePhysicalState

ResourcePhysicalState, Table 24, describes physical state parameters for the resource.

Table 24: ResourcePhysicalState Detail

Name	Type	Notes
Phases	Phases	Phase.
Power	Power	Power.

2.3.22. ResourceStatus

ResourceStatus, Table 25, describes electrical status at the resource connection.

Table 25: ResourceStatus Detail

Name	Type	Notes
Current	Current	Current.
Phases	Phases	Phase
Power	Power	Power.
Status	boolean	Indicates if resource is active (true) or inactive (false).
Voltage	Voltage	Voltage.

2.3.23. SupervisoryControlSignal

Supervisory control signal provided to LocalController, Table 26.

Table 26: SupervisoryControlSignal Detail

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

2.3.24. Tender

Tender data structure exchanged by TransactiveAgent during negotiation, Table 27.

Table 27: Tender Detail

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

2.3.25. TenderComponent

A component of a tender, Table 28.

Table 28: TenderComponent Detail

Name	Type	Notes
priceCurve	PriceCurve	Price curve for this time reference.
timeReference	int	Time reference for this tender component.

2.3.26. Transaction

A transaction used to establish a price between TransactiveAgents, Table 29.

Table 29: Transaction Detail

Name	Type	Notes
accept	boolean	Accept the last quote.

2.3.27. Voltage

A complex vector of Voltage and a string enumeration of the phases, Table 30.

Table 30: Voltage Detail

Name	Type	Notes
voltageByPhase	ComplexNumber	Voltage by phase.

2.3.28. TimeReference

A time reference, in Universal Time Coordinated (UTC), Table 31.

Table 31: TimeReference Detail

Name	Type	Notes
t	int	Time reference.

2.3.29. GridNodeId

An identifier representing a grid node.

2.3.30. 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 combinations to represent phasing in power line segments, circuits and transformers. For example, 3-Phase Electric Power with a neutral would be "ABCN". Two-phase residential power is typically "S12N". Details are in Table 32.

Table 32: Phases Detail

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

2.3.31. StorageType

An enumeration that defines the energy storage characteristics of an instance of the Generator Class, Table 33.

Table 33: StorageType Detail

Name	Type	Notes
none	enum	This value indicates that this instance of the Generator Class models a device that does not produce energy from storage.
electricalStorage	enum	This value indicates that this instance of the Generator Class models a device that produces electricity from storage.
thermalStorage	enum	This value indicates that this instance of the Generator Class models a device that produces thermal energy from storage.

2.3.32. SupplyStatusType

This enumeration, Table 34, indicates if the load is presently curtailed and if curtailed is in compliance with the curtailment request received.

Table 34: SupplyStatusType Detail

Name	Type	Notes
supplyInactive	enum	This generator is not presently operating
supplyRequestPending	enum	A request has been received and is pending.
supplyCompliant	enum	The generator operation is compliant with the last request.
supplyNoncompliant	enum	The generator is not compliant with the last request.

2.3.33. LoadStatusType

This enumeration provides the present overall state of the load, Table 35.

Table 35: LoadStatusType Detail

Name	Type	Notes
NA	enum	Not applicable
eligible	enum	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	enum	The load is ineligible for curtailment since it has been locked.
loadOverridden	enum	An external process has set this override attribute prohibiting curtailment.
lostCommunication	enum	The load presently cannot be accessed.
maxCurtailQueue	enum	The load is in curtailment and presently being timed for the maximum curtailment time.
maxCyclesThisPeriod	enum	The load has been cycled the maximum number of times this period.
minCurtailQueue	enum	The load is in curtailment and presently being timed for the minimum curtailment time.
pointNotConsumingEnergy	enum	The load is ineligible for curtailment since the point associated with the load is already shut off and not consuming any energy.
pointUnreliable	enum	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	enum	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.

Name	Type	Notes
releaseQueue	enum	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	enum	The load has recently been unlocked. It will analyze all conditions and set its present eligibility state after analysis completes.

3. 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:

- The platform on which the experiment is running has a class model based on the Transactive Energy Abstract Component Model;
- The interfaces of the model are implemented as a publish-subscribe (pub-sub) messaging pattern or an equivalent; and
- The interfaces provide at least the specified data from the component model interface definitions.
- Note that whether the model components can be combined from different sources on any given platform can't be guaranteed by the abstract model. But if they are compatible the experiments can be composed.

3.1. Base TE Experiment Scenario diagram

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

- Initialization of all components by the Experiment Manager; and
- Three parallel sequences that continue until experiment ends:
    - Physical: represents the timing needed to perform multiphysics power and energy simulation;
    - Logical Controller: represents the timing needed to perform supervisory and local control of resources; and
    - Transactive: represents the timing needed to perform a periodic transactive sequence resulting in a pricing model for the duration of the transactive step. This includes a "settle" loop for performing market/participant convergence on price.

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

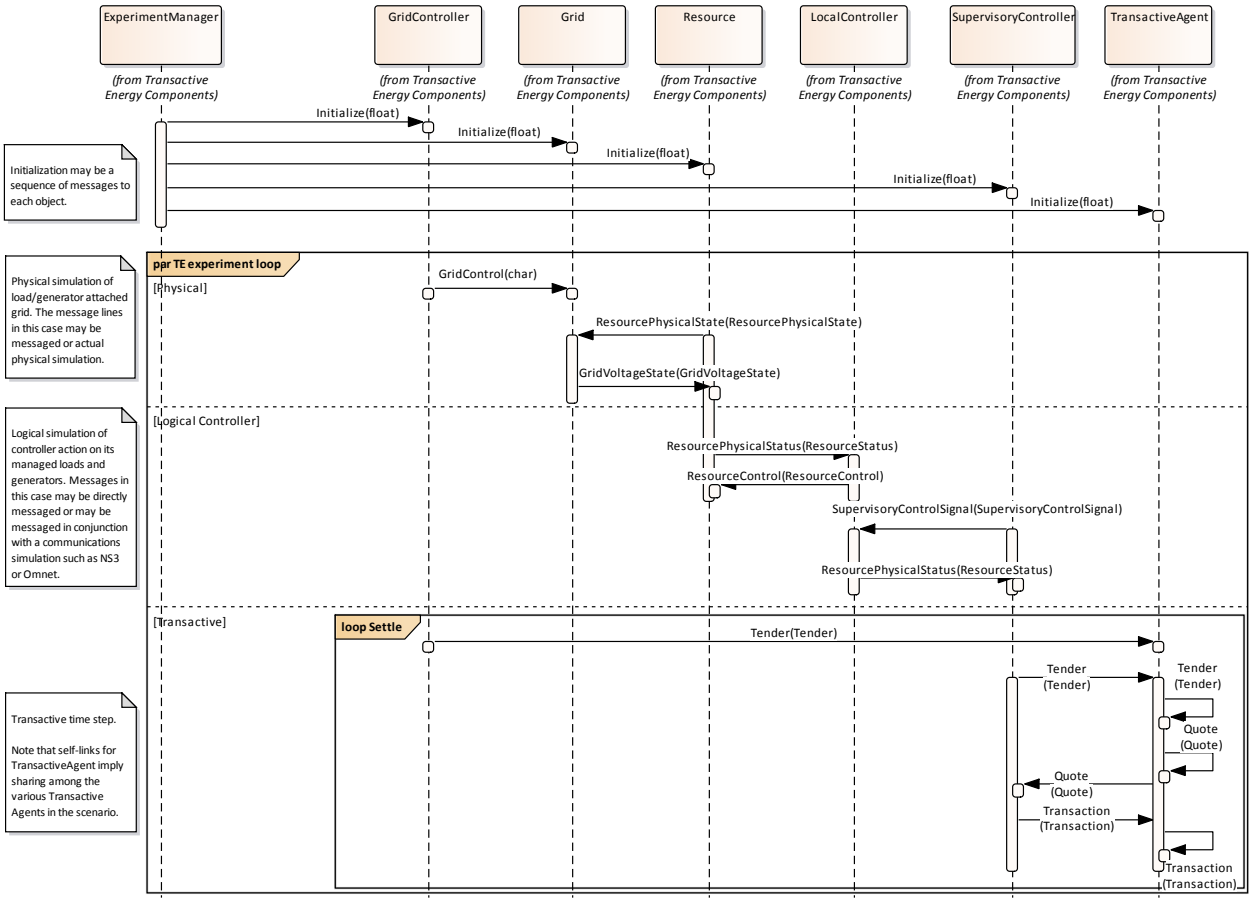

Figure 6: Base TE Experiment Scenario

3.2. Settle

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

3.3. TE experiment loop

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

4. Beta Use Case

A Beta Use Case was provided by PNNL contributors to this modeling effort. The use case itself was realized completely within GridLAB-D and is available from [1].

Figure 7 illustrates the 30 house 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.

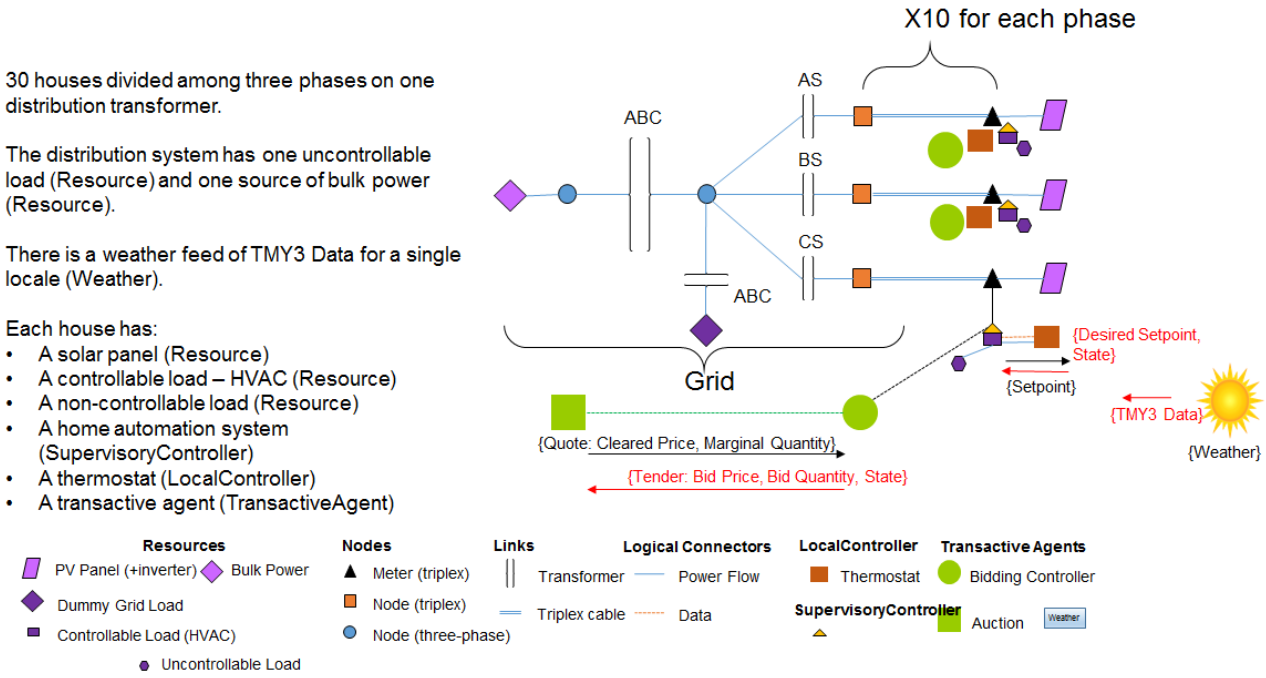

Figure 7: Beta Use Case

The balance of this section will illustrate the Beta Use Case and its components, including the traceability to the model. The model exclusively uses SI units. However, GridLAB-D uses predominantly Imperial units.

4.1. Beta Use Case diagram

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

Figure 8 shows a UML package diagram that is a map to how the various parts of the use case are illustrated.

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

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

The details of these components can be inspected in the model UML in the reference [1].

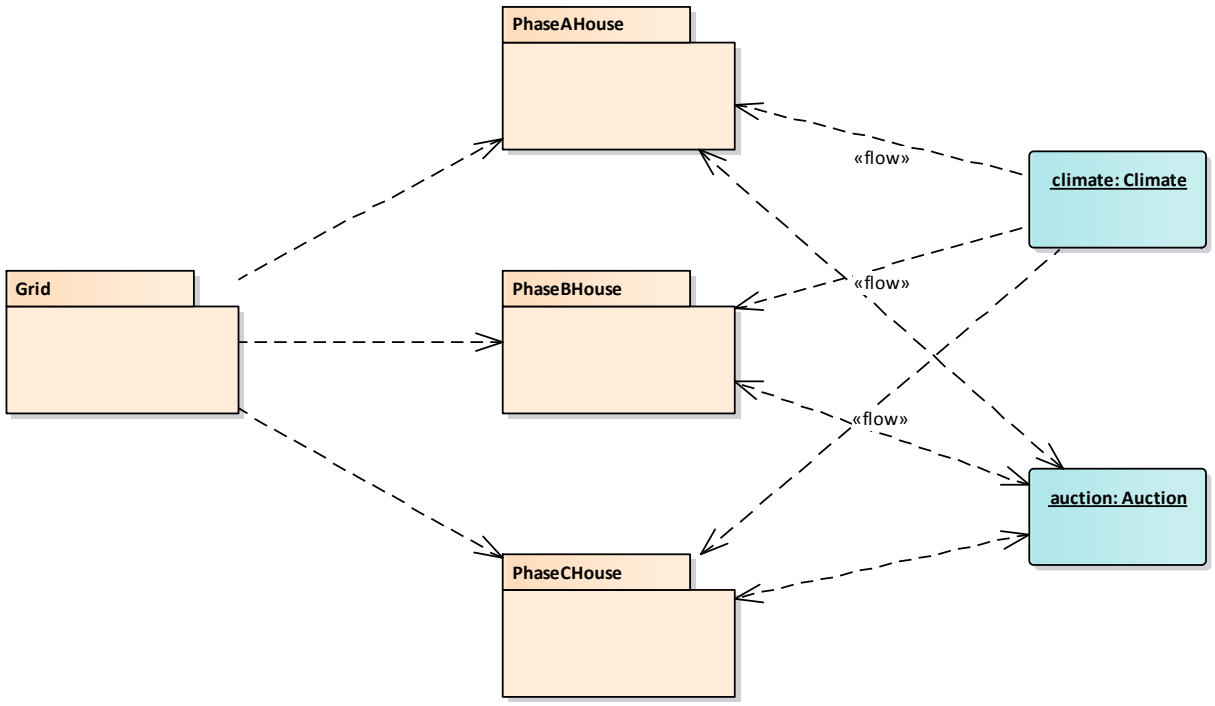

Figure 8: Beta Use Case

4.2. Common Component Inheritance from GridLAB-D

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

4.2.1. Common Component Inheritance diagram

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

Note that some parts of the GridLAB-D classes are not separate and distinct classes but do correspond to the functions of the component model. In these cases, pseudo classes are identified in the diagram (in yellow — Null_Controller, ZIPLoad_controller, and HVAC_controller) to make the inheritance map complete.

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

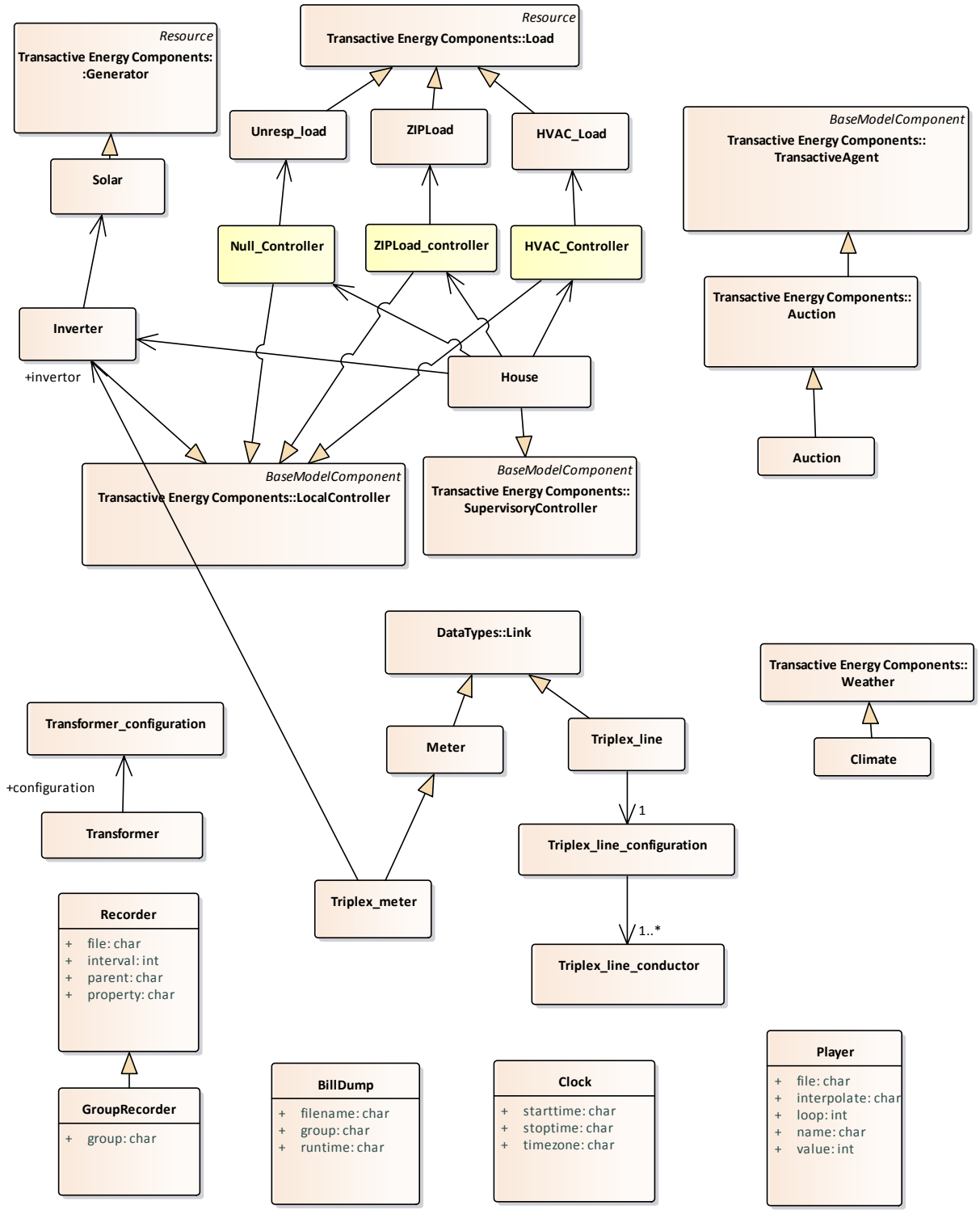

Figure 9: GridLAB-D Common Component Inheritance

4.4. Grid

This package describes the Beta Use Case grid model.

4.4.1. Grid-Part 1 diagram

Figure 10 illustrates the Beta Use Case grid model. This is part 1 of 2.

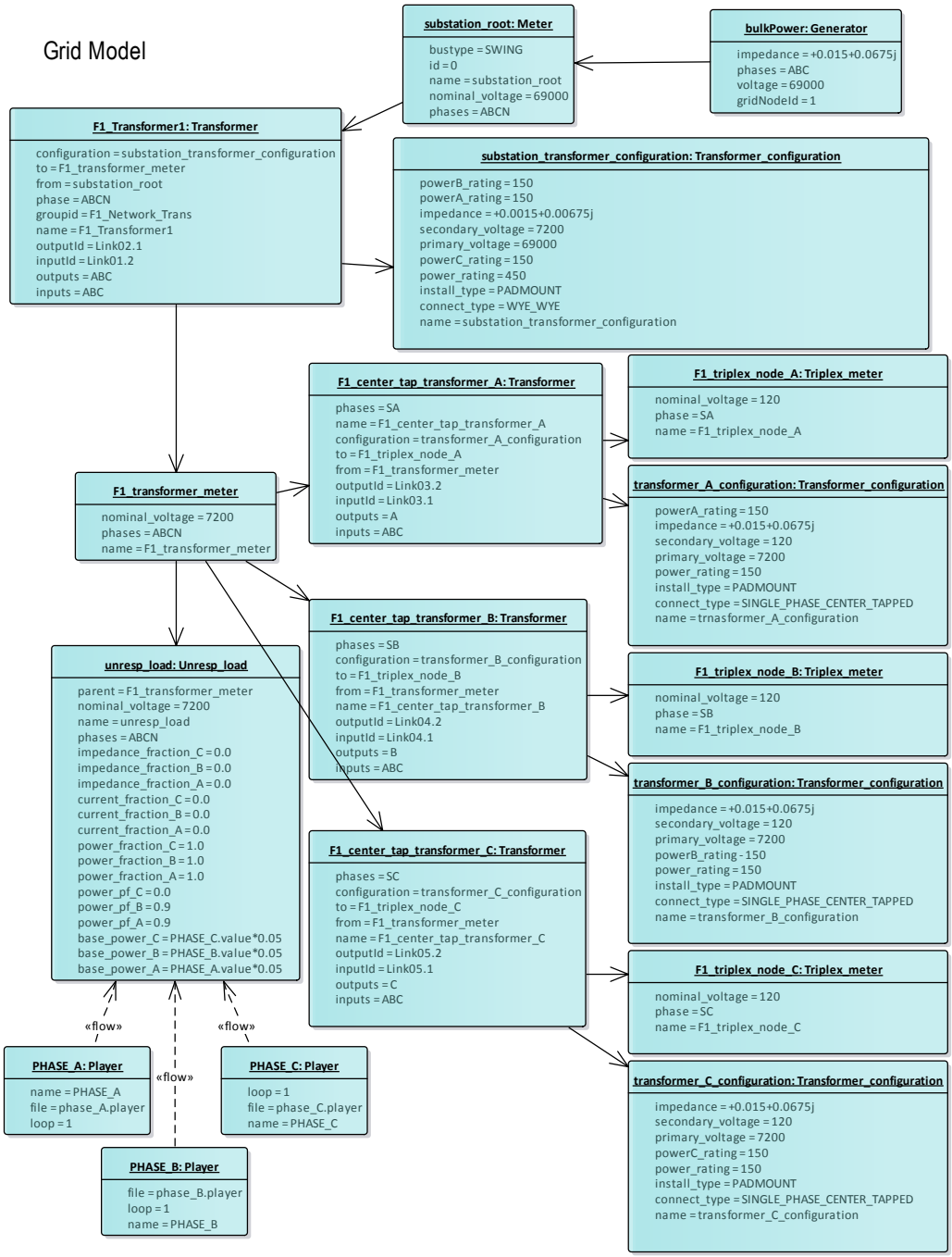

Figure 10: Grid-Part 1

4.4.2. Grid Part 2 diagram

Figure 11 illustrates the Beta Use Case grid model. This is part 2 of 2.

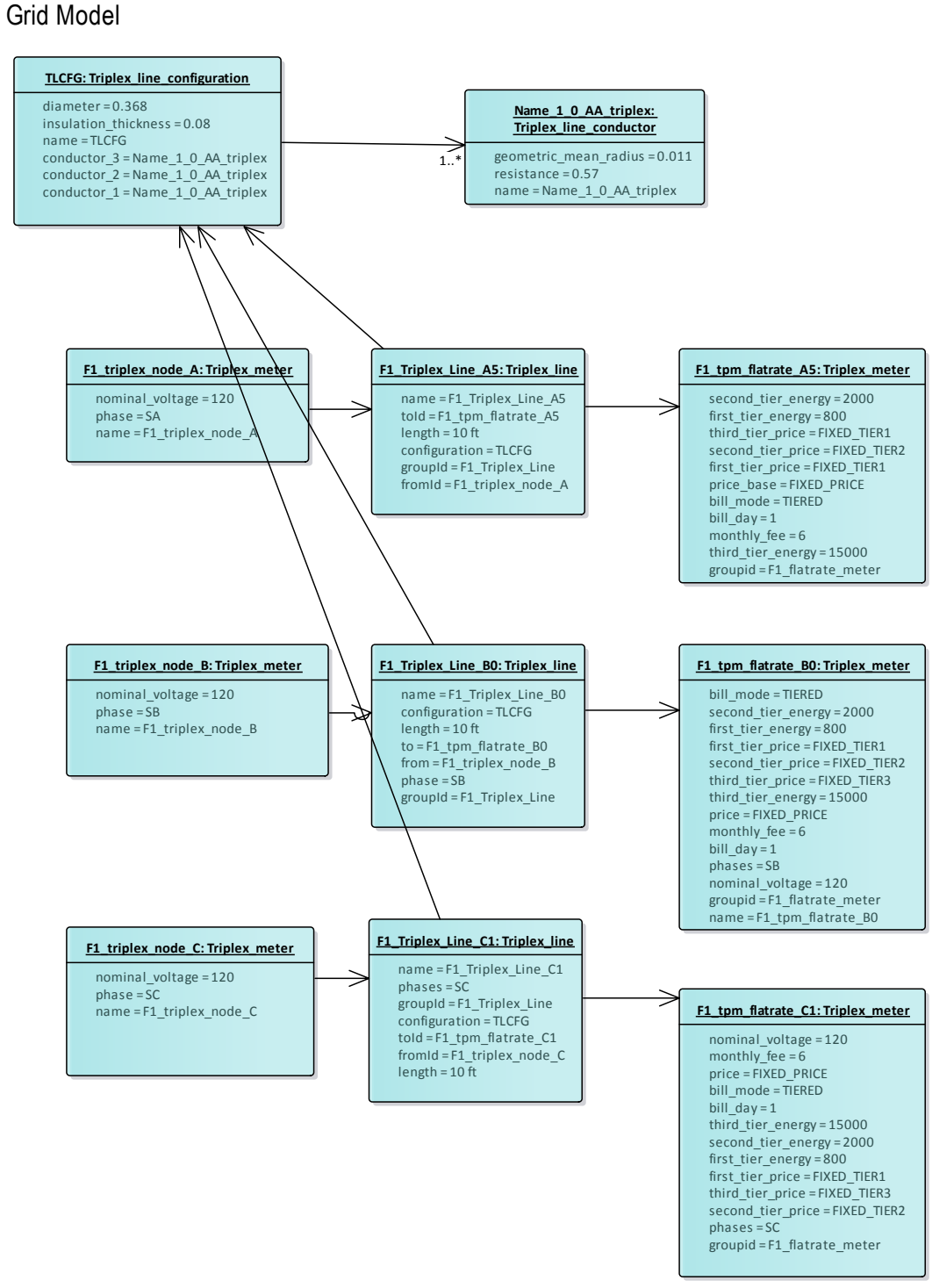

Figure 11: Grid Part 2

4.5. 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.

4.5.1. PhaseAHouse diagram

Houses connected to Phase A are shown in this diagram, Fig. 12.

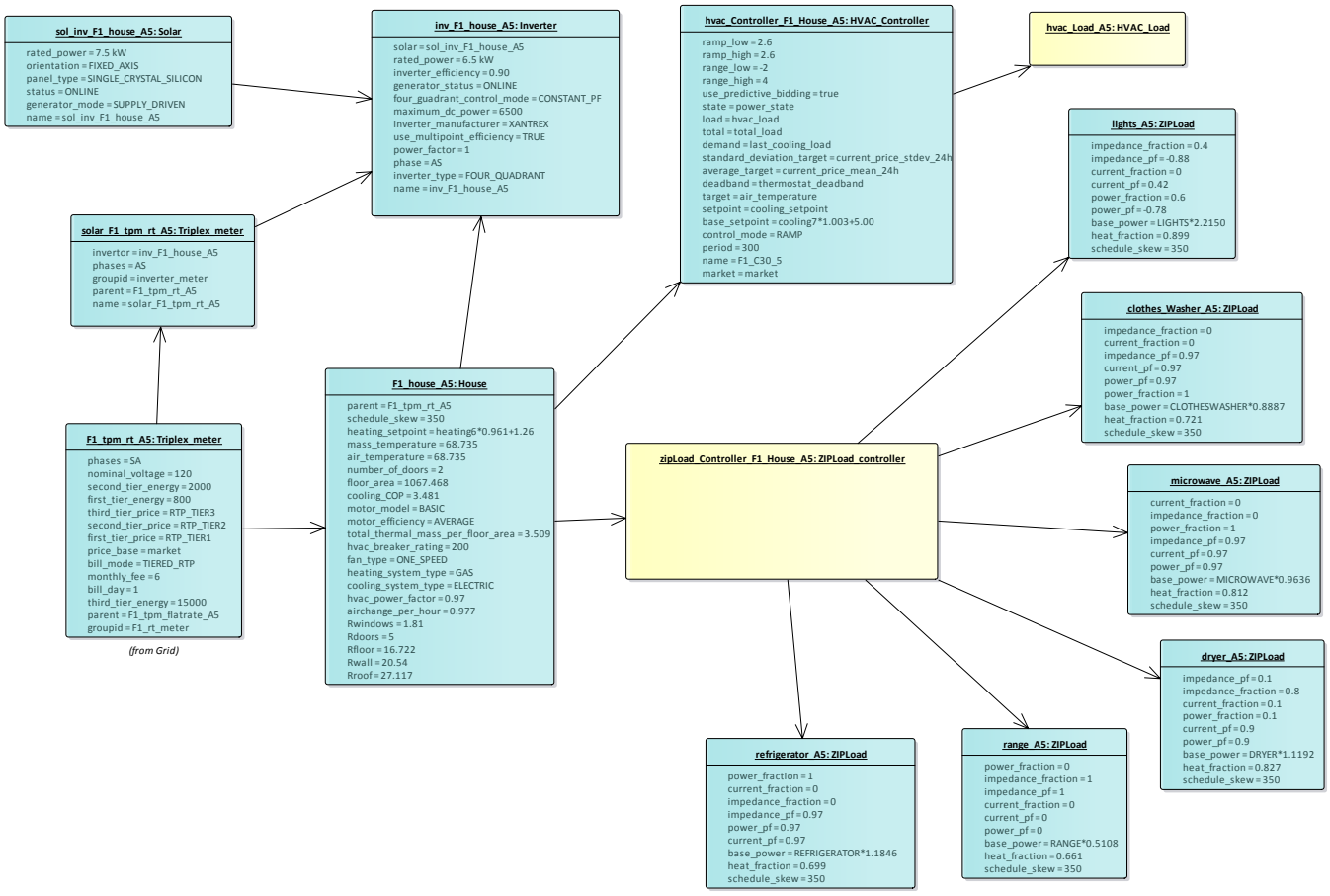

Figure 12: PhaseAHouse

4.6. 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.

4.6.1. PhaseBHouse diagram

Houses connected to Phase B are shown in this diagram, Fig. 13.

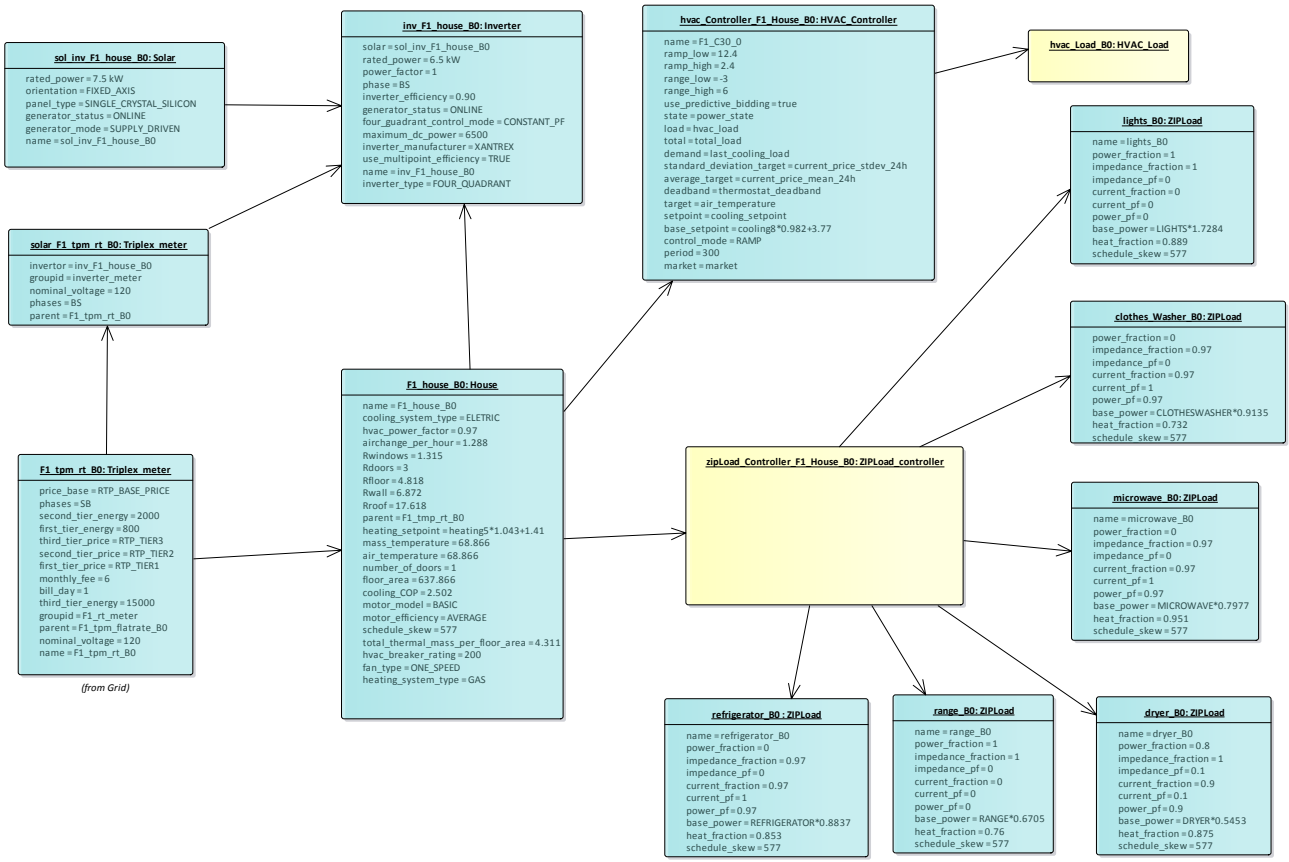

Figure 13: PhaseBHouse

4.7. 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.

4.7.1. PhaseCHouse diagram

Houses connected to Phase C are shown in this diagram, Fig. 14.

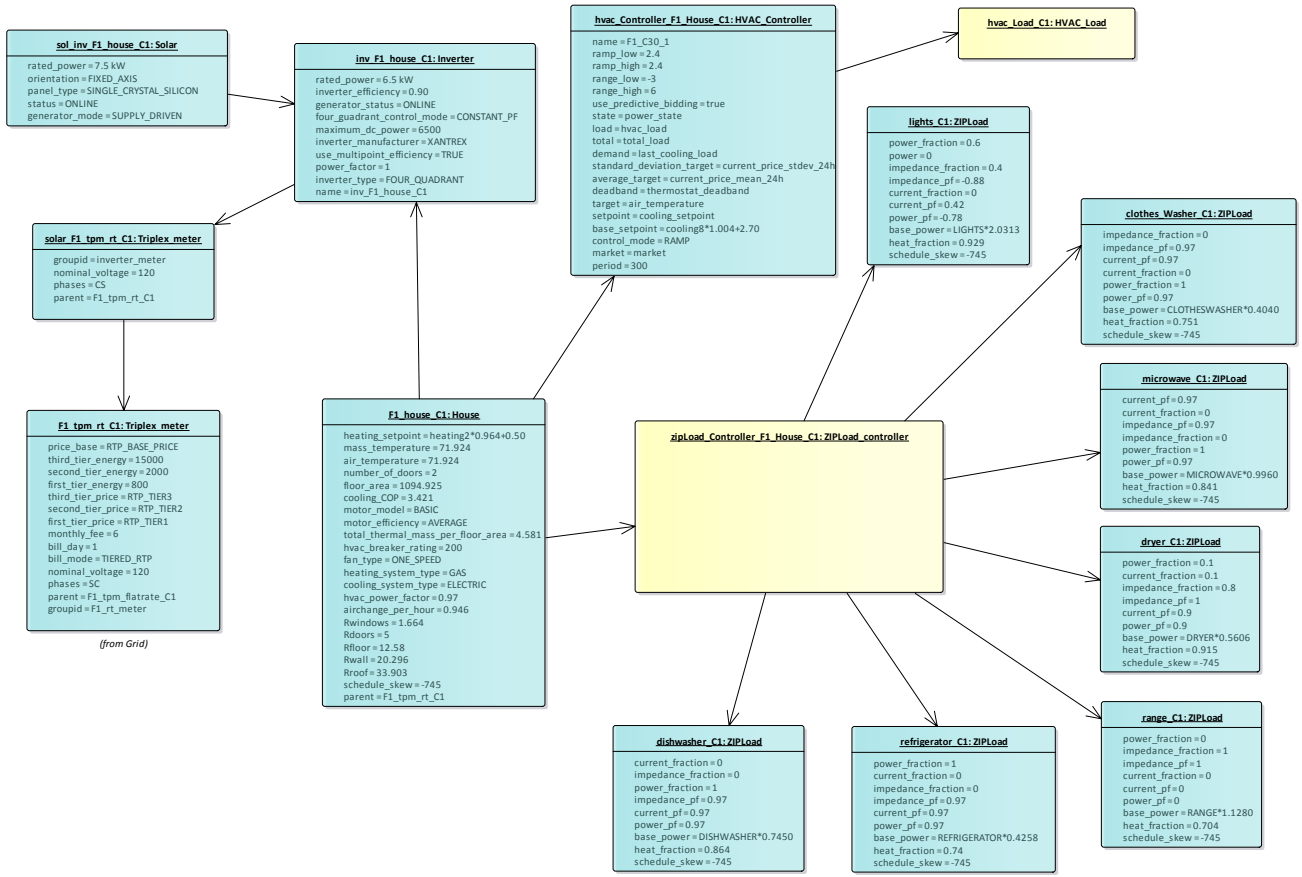

Figure 14: PhaseCHouse

5. Composite and Extended Classes

Because the model uses publish and subscribe technology and is based on interfaces, implementing components that realize the interfaces is all that is necessary to be composed into a model simulation.

Composite classes allow 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.

5.1. Composite and Extended Classes diagram

This diagram, Fig. 15, illustrates how components can be combined and/or extended for use in experiments of the Transactive Energy Abstract Component Model.

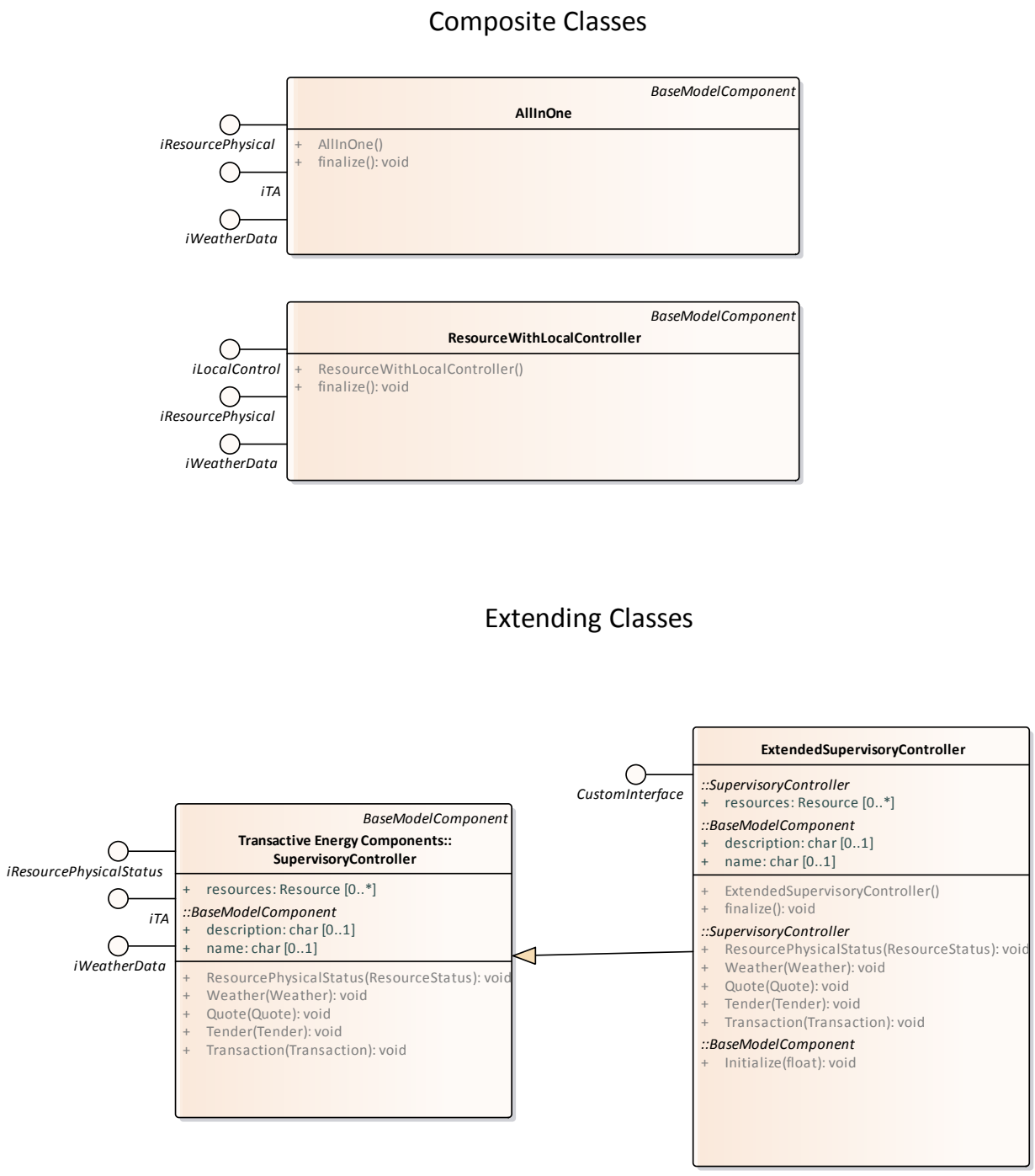

Figure 15: Composite and Extended Classes

5.2. AllInOne

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

5.3. 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.

5.4. ResourceWithLocalController

Contains Resource with Local Controller.

5.5. CustomInterface

A custom extension interface for one of the core components.

References

[1] NIST (2018) TE Challenge Component Model: retrieved from https://github.com/usnistgov/TEChallengeComponentModel

[2] Universal CPS Environment for Federation (UCEF) project collaboration site. Retrieved from https://pages.nist.gov/ucef/

[3] Pacific Northwest National Labs (PNNL) Transactive Energy Simulation Platform (TESP). Retrieved from http://tesp.readthedocs.io/en/latest/

[4] 110th Congress Public Law 140 (2007), *Energy Independence and Security Act Of 2007*, retrieved from https://www.gpo.gov/fdsys/pkg/PLAW-110publ140/html/PLAW-110publ140.htm

[5] NIST Special Publication 1108r3, (2014), *NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0*, http://dx.doi.org/10.6028/NIST.SP.1108r3

[6] GridWise Architecture Council (2015), *GridWise Transactive Energy Framework Version 1.0* , available at: http://www.gridwiseac.org/pdfs/te_framework_report_pnnl-22946.pdf

[7] Holmberg, D., Burns, M. et. al., "NIST TE Challenge Phase II Report", NIST Special Pub 1900-603, 2018. https://doi.org/10.6028/NIST.SP.1900-603

[8] GridLAB-D, PNNL, retrieved from https://www.gridlabd.org

[9] Holmberg, D., Hardin, D, Cunningham, R, Melton, R, Widergren, S, (2016) SGIP Technical Paper, *Transactive Energy Application Landscape Scenarios*, available at: https://sepapower.org/resource/transactive-energy-application-landscape-scenarios/

[10] PNNL-SA-113294 (2015), *Transactive Valuation Methodology Insights*, available at: http://www.gridwiseac.org/pdfs/workshop_20150929/pnnl_sa_113294.pdf.

[11] NREL Typical Meteorological Year TMY(3), http://rredc.nrel.gov/solar/old_data/nsrdb/1991-2005/tmy3/

[12] ASHRAE (2016). Facility Smart Grid Information Model (FSGIM), ANSI/ASHRAE Standard 201-2016.

Acronyms

The following acronyms are presented as a ready reference to the intended meaning of their use in the text of this document. It is recognized that within various technical domains, many of these terms and acronyms have multiple meanings. The intent is to provide clarity for the interpretation of this framework and not to make a definitive statement about the "universal" definition of the terms and acronyms.

Selected acronyms used in this document are defined below.

Acronym	Expansion
3D	Three dimensional
DOE	Department of Energy
EISA	Energy Independence and Security Act
FSGIM	Facility Smart Grid Information Model
HVAC	Heating, Ventilating, and Air Conditioning
IEEE	Institute of Electrical and Electronics Engineers
NIST	National Institute of Standards and Technology
PNNL	Pacific Northwest National Labs
TA	Transactive Agent
TE	Transactive Energy
TESP	Transactive Energy Simulation Platform
UCEF	Universal CPS Environment for Federation