1 Message Composition & Services

Energy Interoperaion relies on two other standards, Energy Market Information eXchange (EMIX) and [WS-Calendar] to express intents.

- EMIX describes price and product for electricity markets.
- WS-Calendar communicates schedules and sequences of operations.
- Energy Interoperaion uses the vocabulary and information models defined by those specifications to describe many of the services that it provides.

1.1 WS-Calendar in Energy Interoperaion

[WS-Calendar] defines how to use the semantics of the enterprise calendar communications within service communications. Energy Interoperaion is conformant with the [WS-Calendar] specification for communicating duration and time to define a Schedule. [WS-Calendar] itself extends the well-known semantics of [RFC5545]. The communication of a commonly understood Schedule is essential to Energy Interoperaion.

Energy Interoperaion also relies on [EMIX], which defines schedules and types conforming to [WS-Calendar]. Energy Interoperaion is conformant with the [WS-Calendar] specification for communicating duration and time to define a Schedule.

1.1.1 Schedule Semantics from WS-Calendar (Non-Normative)

Without an understanding of certain terms defined in [WS-Calendar], the reader may have difficulty achieving complete understanding of their use in this standard. The table below provides summary descriptions of certain key terms from that specification. This specification does not redefine these terms; they are listed here solely as a convenience to the reader.

<table>
<thead>
<tr>
<th>WS-Calendar Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>In [iCalendar], the primary information structure is a Component, also referred to as a &quot;vobject.&quot; A Component is refined by Parameters and can itself contain Components. Several RFCs have extended iCalendar by defining new Components using the common semantics defined in that specification. In the list below, Interval, Gluon, and Availability are Components. Duration, Link, and Relationship are Parameters. A Sequence is set of Components, primarily Intervals and Gluons, but is not itself a Type.</td>
</tr>
<tr>
<td>Duration</td>
<td>Duration is the length of time for an event scheduled using iCalendar or any of its derivatives. The [XCAL] Duration is a data type using the string representation defined in the iCalendar (RFC3658) Duration.</td>
</tr>
<tr>
<td>Interval</td>
<td>The Interval is a single discrete segment, an element of a Sequence, and expressed with a Duration. The Interval is derived from the common calendar Components. An Interval is part of a Sequence.</td>
</tr>
<tr>
<td>Sequence</td>
<td>A set of Intervals with defined temporal relationships. Sequences may have gaps between Intervals, or even simultaneous activities. A Sequence is re-locatable, i.e., it does not have a specific date and time. A Sequence may consist of a single Interval, and can be scheduled by scheduling that single Interval in that Sequence.</td>
</tr>
<tr>
<td>Gluon</td>
<td>A Gluon influences the serialization of Intervals in a Sequence, through inheritance and through schedule setting. The Gluon is similar to the Interval, but has no service or schedule effects until applied to an Interval or Sequence.</td>
</tr>
<tr>
<td>Artifact</td>
<td>The placeholder in a Component that holds that thing that occurs during an Interval. [EMIX] Product Descriptions populate Schedules as Artifacts inside Intervals. In Streams, this specification refers to the Payload conveyed by an Interval.</td>
</tr>
<tr>
<td>Link</td>
<td>A reference to an internal object within the same calendar, or an external object in a remote system. The Link is used by one [WS-Calendar] Component to reference another.</td>
</tr>
<tr>
<td>Relationship</td>
<td>Links between Components.</td>
</tr>
<tr>
<td>Availability</td>
<td>Availability in this specification refers to the Vavailability Component, itself a collection of recurring Availability parameters each of which expresses set of Availability Windows.</td>
</tr>
</tbody>
</table>
Normative descriptions of the terms in the table above are in [WS-Calendar].

### 1.1.2 Schedules and Inheritance

Nearly every response, every event, and every interaction in Energy Interoperation (with the exception of all single interval TeMIX profile interactions) can have payloads with values that vary over time, i.e., it is described using a sequence of intervals. Many communications, particularly in today’s retail market, involve information about or a request for power delivered over a single interval of time. Simplicity and parsimony of expression must coexist with complexity and syntactical richness.

The simplest power description in [EMIX] is Transactive power. The simplest demand response is to reduce power. The power object in EMIX can include specification of voltage, and Hertz and quality and other features. There are market interactions where each of those is necessary. Reduced to its simplest, though, the EMIX Power information consists of Power Units and Power Quantity: as in

<table>
<thead>
<tr>
<th>Units</th>
<th>KW</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>

**Figure 1:** Basic Power Object from EMIX

At its simplest, though, WS-Calendar expresses repeating intervals of the same duration, one after the other, and something that changes over the course of the schedule

<table>
<thead>
<tr>
<th>Start: 8:00</th>
<th>Duration: 1Hour</th>
<th>Quantity: 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration: 1Hour</td>
<td>Quantity: 10</td>
<td></td>
</tr>
<tr>
<td>Duration: 1Hour</td>
<td>Quantity: 15</td>
<td></td>
</tr>
<tr>
<td>Duration: 1Hour</td>
<td>Quantity: 25</td>
<td></td>
</tr>
<tr>
<td>Duration: 1Hour</td>
<td>Quantity: 10</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 2:** WS-Calendar Partition, a simple sequence of 5 intervals

The WS-Calendar specification defines how to spread an object like the first over the schedule. The information that is true for every interval is expressed once only. The information that changes during each interval, is expressed as part of each interval.

<table>
<thead>
<tr>
<th>Units</th>
<th>KW</th>
<th>Start: 8:00</th>
<th>Duration: 1Hour</th>
<th>Quantity: 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Duration: 1Hour</td>
<td>Quantity: 10</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Duration: 1Hour</td>
<td>Quantity: 15</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Duration: 1Hour</td>
<td>Quantity: 25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Duration: 1Hour</td>
<td>Quantity: 10</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3:** Applying Basic Power to a Sequence

Many communications communicate requirements for a single interval. When expressing market information about a single interval, the market object (Power) and the single interval collapse to a simple model:

**Figure 4:** Simplifying back to Power in a Single Interval

WS-Calendar calls this pattern Inheritance and specifies a number of rules that govern Inheritance. Table 2 summarizes those terms defined in WS-Calendar to describe Inheritance.
that are used in this specification as well. This specification does not redefine these terms; they are listed here solely as a convenience to the reader.

### Table 2: WS-Calendar Semantics: Inheritance

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lineage</td>
<td>The ordered set of Parents that results in a given inheritance or execution context for a Sequence.</td>
</tr>
<tr>
<td>Inherit</td>
<td>A Child Inherits attributes (Inheritance) from its Parent.</td>
</tr>
<tr>
<td>Inheritance</td>
<td>A pattern by which information in Sequence is completed or modified by information from a Gluon. Information specified in one informational object is considered present in another that is itself lacking expression of that information.</td>
</tr>
<tr>
<td>Bequeath</td>
<td>A Parent Bequeaths attributes (Inheritance) to its Children.</td>
</tr>
</tbody>
</table>

This specification extends the use of Inheritance as defined in WS-Calendar. Most interactions specify a schedule, whether for price Quote or for Demand Response event. These schedules are expressed in Streams (see Section 1.3). Each Interval in the Schedule contains an information payload. Each of these payloads is completed through inheriting information from the Stream as if from a Gluon. The Stream itself inherits information from the context of the interaction, especially from the Market Context, as if from a Gluon.

A Market Context Bequeaths essential information to a Stream, which in turn its information to each Interval in the Stream. This specification uses this pattern of expression throughout.

#### 1.1.3 Availability and Schedules

The WS-Calendar component Availability is used throughout Energy Interoperation. Availability expresses recurring patterns of schedule within a bounded period of time. This specification uses Availability in market definitions and in a number of inter-party commitments and communications. Availability is used to define windows for Demand Response, to define when during a given day a Party may receive requests, and for expressing the desire of a Party to place or remove services from markets.

While the expression of Availability is defined in WS-Calendar, the Committee recommends the informative discussion of Availability found in [Vavailability].

#### 1.1.4 Smoothing Response

Precision of communication and response causes special problems for large collections of entities and systems, as well as for switching of high electrical demand as in substations or with large electric motors. When devices interact at high speeds to change demand, they can create sharp spikes up or down in demand. These spikes can affect other nodes on a grid, cause a grid to crash, or even destroy equipment.

WS-Calendar defines Tolerance as an optional Property of Intervals that expresses allowable imprecision. Tolerance may have up to 5 parameters: Start Before Tolerance, Start After Tolerance, End Before Tolerance, End After Tolerance, and Precision.

For example, Start Before Tolerance may have a value of ten minutes. In the same Interval, Start After Tolerance may have a value of five minutes. Let us further specify that the Interval starts at 3:00 PM with a Duration of two hours. WS-Calendar then has expressed that the recipient begin its response at 3:00 and continue for two hours, but that a response that begins any time between 2:50 pm and 3:05 pm is acceptable.

For convenience, this specification refers to the Tolerance Interval as either the sum of the starting tolerances (Start Before Tolerance and Start After Tolerance) or the sum of the ending tolerances (End Before Tolerance and End After Tolerance).

Because Sequences are constructed of linked intervals expressed as Durations, Tolerance applied only to the Designated Interval in a Sequence can change the interpretation of the entire Sequence. If the Designated Interval begins five minutes late and lasts one hour, then the second Interval, which is anchored by the first, will also begin five minutes late, and so on.
The Smart Grid is a system of systems, and each system provides its respective class of application. Some systems are aggregates of hundreds or thousands of similar systems. Other systems contain many internal systems with their own dependencies and interactions. Still others may consist of a single large system. Each of these represents a different application.

- Applications managing small loads may be required to randomize their start time within Tolerance Interval. Conformance requirements for a deployment must specify how this randomization is demonstrated or evaluated for a particular application.

- Applications internally managing collections of smaller loads may be required to spread the starts and stops of each internal system to produce a load that moves in steps over the Tolerance Interval. Different systems may do this differently. Integrated systems will sequence their internal loads to manage internal cross-dependencies. Less integrated systems may randomize the starts of their internal systems. Conformance for these applications may include a minimum spread of steps or a maximum quantum change of load.

- Applications that front single large loads may be required to gradually ramp between the initial state and the requested response across the Tolerance Interval.

Conformance to these deployment scenarios is outside the scope of this specification.

1.2 EMIX in Energy Interoperation

Energy Interoperation uses EMIX to express the semantics of Power and Energy Markets.

In [EMIX] Product Descriptions define Energy and Power. Product Descriptions are applied to Sequences to create Schedules. Schedules conform to the inheritance pattern defined in [WS-Calendar] to reduce repetition of these descriptive elements. [EMIX] Products include an entire Schedule along with transactional information. [EMIX] Options use Availability to describe market information for the right to acquire Energy during certain periods at specified Rates. TeMIX defines communications for transactions of energy delivered at specified rates over specific intervals.

Each of the elements above is associated with a Market Context. A Market Context may be associated with Standard Terms which may define an overriding set of information for products therein. An [EMIX] Schedule can inherit information from the Standard Terms in a Market just as a WS-Calendar Sequence inherits from a Gluon.

Every Energy Interoperation interaction MAY convey an EMIX Type. Often they convey simplified derivations of [EMIX] types that use conformance and inheritance to reduce to a bare minimum, while still using EMIX semantics.

Energy Interoperation defines Parties which enroll with Counter-Parties. These Parties may then participate directly in energy transactions, using the Semantics from TeMIX. Others enroll as Resources with certain capabilities. Some of these Resources may share detailed capability and response information with their counter-party using the EMIX Resource semantics.

1.2.1 Core Semantics from EMIX

The terms in Table 3 are normatively defined in [EMIX]. Summary descriptions are provided here for the convenience of the reader only.

<table>
<thead>
<tr>
<th>Table 3: EMIX Essential Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EMIX Term</strong></td>
</tr>
<tr>
<td>Item Base</td>
</tr>
<tr>
<td>Schedule</td>
</tr>
<tr>
<td>Product Description</td>
</tr>
</tbody>
</table>
### EMIX Terms

<table>
<thead>
<tr>
<th>EMIX Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMIX Base</td>
<td>The EMIX Base conveys a Schedule populated with Product Descriptions and is intended to express additional market information sufficient to define Products.</td>
</tr>
<tr>
<td>Price Base</td>
<td>The PriceBase conveys a Price, a Relative Price, or a Price Multiplier.</td>
</tr>
<tr>
<td>EMIX Interface</td>
<td>Abstract base class for the interfaces for EMIX Product delivery, measurement, and/or pricing. The PNode and the Service Area are examples of the EMIX Interface.</td>
</tr>
<tr>
<td>Market Context</td>
<td>A URI uniquely identifying a source for market terms, market rules, market prices, etc.</td>
</tr>
<tr>
<td>EMIX Product</td>
<td>A Product Description applied to a Schedule. Using the Gluon / Sequence pattern of inheritance, there may be a nearly complete Product Description in the element that acts as a Gluon, and only elements that change in each interval.</td>
</tr>
<tr>
<td>EMIX Option</td>
<td>A Type of Product in which for a defined price, a party agrees to make Product available during a schedule (Availability) to be delivered at the counterparty's request, in accord with agreed upon terms and at an agreed upon price.</td>
</tr>
<tr>
<td>Transactive State</td>
<td>An indicator included in EMIX Base derived types to aid in processing. The enumerated Transactive States are: Indication Of Interest, Tender, Transaction, Exercise, Delivery, Transport Commitment, and Publication.</td>
</tr>
<tr>
<td>Terms</td>
<td>Terms are used in EMIX to describe when and how a product is available. Minimum Notification Duration, Maximum Run Duration, and Minimum Remuneration per Event are all Terms.</td>
</tr>
<tr>
<td>Service Area</td>
<td>The Service Area is the only Interface defined for all derived schemas. The Service Area expresses locations or geographic regions relevant to price communication. For example, a change in price for a power product could apply to all customers in an urban area.</td>
</tr>
<tr>
<td>Power</td>
<td>The EMIX Power schema defines products related to the exchange of Electrical Power using the EMIX semantics.</td>
</tr>
<tr>
<td>Resource</td>
<td>The EMIX Resource schema defines the capabilities that a node has to deliver Power products.</td>
</tr>
<tr>
<td>Ancillary Service</td>
<td>Ancillary Services are typically products provided by a Resource contracted to stand by for a request to deliver changes in power to balance the grid on short notice.</td>
</tr>
</tbody>
</table>

The terms in Table 3 are defined normatively in EMIX and nothing in this specification changes or overrides those definitions.

### 1.2.2 Putting EMIX in Context

EMIX specifies that information that does not change can be summarized using standard Terms associated with a Market Context.

### Table 4: EMIX Market Context

<table>
<thead>
<tr>
<th>Expectations and Contexts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Market Context</td>
<td>Defines the product, performance expectations and rules for interactions. All Events, Signals, and Transactions occur within a market context. A Market Context acts as a Gluon for all sequences described in the EI Types. Market Contexts are described using the semantics of EMIX Standard Terms.</td>
</tr>
<tr>
<td>Availability</td>
<td>Describes when a Resource is available to respond relative to a particular VTN and Market Context.</td>
</tr>
<tr>
<td>Market Expectations</td>
<td>Market Expectations are associated with a Market Context and consist of a number of Rule Sets.</td>
</tr>
<tr>
<td>Standard Terms</td>
<td>Standard Terms apply to all transactions in a Market Context. When they are conveyed as Standard Terms, they do not need to be repeated in individual interactions. A product references a Market Context and all Standard Terms associated with that Market Context.</td>
</tr>
<tr>
<td>Granularity</td>
<td>Granularity is the units of time used in operating a market, i.e., a market with a granularity of one hour transacts power in one hour increments. A One hour market is for one-hour purchases of Power with each interval in a one hour modulo offset from the beginning of the business schedule.</td>
</tr>
</tbody>
</table>
The terms in Table 4 are defined normatively in EMIX and nothing in this specification changes or overrides those definitions.

1.3 Streams: Adaptations of WS-Calendar for Energy Interoperation

Streams use WS-Calendar Sequences to convey a time sequence of prices, usage, demand, response, or anything else that varies over time. Streams are used both for projections of the future and for reports about the past; event signals and reports are each instances of Streams.

WS-Calendar specifies that Sequences that describe a Service be expressed as Duration within each Interval, Temporal Relations between those intervals, and a single Start or End time for the Sequence. WS-Calendar specifies that each Interval have a unique identifier (UID). WS-Calendar further specifies that each Interval include a Temporal Relation, either direct or transitive, with all other Intervals in a Sequence. A Temporal Relation consists of the Relationship, the UID of the related Interval, and the optional Gap between Intervals.

[WS-Calendar] defines a Partition as a Sequence of consecutive Intervals.

All Streams follow the Gluon-Sequence pattern from WS-Calendar, i.e., the Stream acts as a Gluon that optionally contains a degenerate Sequence. Information valid for the entire stream is indicated in the Gluon, i.e., external to the Intervals of the Sequence. Only information that changes over time is contained within each interval. This changing information is referred to herein as the Payload.

![Figure 5: Stream as Gluon and Sequence](image)
For example, an Event establishes in a context specified by Enrollment and each Signal arises within a Market Context and within that Event Base. The information contained in the Event Base MAY inherit information in the Market Context as an Interval or Gluon inherits information from a Gluon. WS-Calendar calls this the lineage of the information.

That Market Context may include Standard Terms, Product Description, Time Zone Identifier (TZID), and Simple Level Definition. The Market Context enters the Lineage (as described in WS-Calendar) of the Schedule as if the Market Context were contained in a Gluon. Product Description, TZID, Program Definition, Terms, et al. can be inherited in this manner. Again, following the WS-Calendar inheritance pattern, each Interval in the Sequence inherits from the Lineage described above.

### 1.3.1 Information Model for Streams

![UML Class Diagram of abstract StreamBase class](image)

**Figure 6:** UML Class Diagram of abstract StreamBase class

#### 1.3.2 Conformance of Streams to WS-Calendar

If it is necessary to process a Stream through standard Calendar communications, the Stream's GUID is the key and the Stream is processed as if a Gluon. All Sequence information MAY remain internal to that Gluon. If it is necessary to instantiate Interval in the Sequence as a WS-Calendar Interval, the GUID for each is derived by appending the Sequence ID to the Stream’s GUID.

### 1.3.2.1 Stream expression of Intervals expressed as Durations

While conformant communications can include anything expressible in [WS-Calendar], this specification further defines standard profiles of Sequences and Intervals for use in Streams.
Streams describe Partitions. Within a Stream expressed using Durations, a virtual UID for each Interval MAY be constructed by concatenating the Stream Identifier, which may include the identity of the source or recipient, and a sequence number. Within a Stream, this UID can be expressed within each interval by the sequence number alone.

If the Designated Interval in a Sequence within a Stream omits a Temporal Relationship, then all Intervals in the Sequence MAY NOT include a Temporal Relation. Such intervals are sorted by increasing sequence number (expressed in the UID), and each Interval is treated as if it contained an implied FinishToStart relation to the next Interval with a Gap of zero Duration.

Partitions expressed in this way consist of Intervals containing only a Sequence Number, the Duration of the Interval (if not inherited), and the Market Signal Payload. The effect of this is that Stream Intervals are ordered as a Partition in order of increasing UID.

WS-Calendar inheritance defines a Lineage whereby Intervals inherit information from Gluons. In Energy Interoperation, Streams are contained within larger messages. A Stream MAY inherit information from its containing message as if from a Gluon. A Stream-derived Type MAY contain information external to the Sequence. This external information inherits acts as if it were a Gluon; it both MAY inherit from the containing message, and Bequeath information to the Designated Interval in the Stream.

The first (in time and in sequence number) Interval in the Sequence in a Stream is the Designated Interval unless another Interval is explicitly so designated in the Stream Event. Signals, Reports, and many other messages use this pattern of expression. For example, the Active Period of an Event Bequeaths its start date and time to an Event Signal which Bequeaths that to the Designated Interval in the sequence. These terms are defined below.

1.3.2.2 Observational Data expressed as Streams

Observed information may be best communicated as raw data without interpretation. A single set of Observations may be re-purposed or re-processed for multiple uses. For example, a measurement recorded at 3:15 may be a point in both a 5 minute series and a 15 minute series. Observational data may have known errors that can be lost in processing. Low-end sensor systems may not update instantly. For example, a reading taken at 4:30 may be known to actually have been recorded at 4:27. Streams expressing a series of observations MAY use the date and times rather than the duration as their primary temporal element.

When the boundaries of Intervals in a Stream are expressed with Date and Time, then all Intervals in that Sequence SHALL be expressed with a Date and Time and that boundary selected SHALL be the Same, i.e., all Intervals MAY be expressed with a Begin Date and Time OR with an End Date and Time. For observations, use the End Date and Time.

Within a Stream expressed using Dates and Times, a virtual UID for each Interval MAY be constructed by concatenating the Signal Identifier, the PartyID (which may be the VEN ID), and the Date and Time. Within an Observational Stream, this UID can be expressed within each interval by the Date and Time alone. Intervals in a Sequence expressed this way are treated as if each contains an implied FinishToStart relation to the next Interval with a Gap of zero duration. The Duration of each Interval can be computed by using the Date(s) and Time(s) of adjacent Intervals.

1.3.3 Payload Optimization in Streams

As defined in WS-Calendar and in EMIX, each Interval in a Sequence potentially contains any artifact that inherits/extends the EMIX Product Description Type as a payload. As used in Streams, the EMIX Artifact is expressed once or inherited from the Market Context. Each Interval in a Stream expresses only the common subset of facts that varies within the context of the Stream. For efficient communication and processing, Streams use these explicit processing rules:

1. Unless each interval includes a full EMIX payload, each Interval in a Stream expresses defined subset of the payload that varies over time.
2. Each Interval in a Stream uses the same payload subset as all other intervals in that stream.
All streams in this specification share a common Payload base. This commonality is derived from the commonality of a request for performance (Signal), a report of performance (Report and Delivery), projections of performance (Projection), and a baseline of performance (Baseline).

![Payload Base Diagram]

**Figure 7:** Payload Base

### 1.3.4 Other elements in Stream Payloads

It may be necessary to qualify information about intervals in the future. The element Interval Qualification extends the WS-Calendar Property. [All Intervals have a collection of Properties]. Energy Interoperation uses Qualifications to indicate the originator’s indications as to how the sender should rely on the information in the Payload.

Qualifications MAY be used in Quotes, in Load and Response projections, and in Observations. They MAY NOT be used in other transactive states.

It may be necessary to qualify measurements delivered in a report. Devices have known accuracies. Several Measurements MAY be added together to create a single quantity. To support these uncertainties different payloads are defined for different services.

Each use of streams in Energy Interoperation, Signals, Baselines, Reports, and Delivery, is discussed below. All four payloads are shown together in Figure 8: Comparing Payloads for Signals, Baselines, Reports, and Delivery.
1.4 Applying EMIX and WS-Calendar to a Power Event

Consider the event in Figure 9. This event illustrates the potential complexity of marshaling a load response from a VEN, perhaps a commercial building.
Figure 9- Demand Response Event and associated Streams

Note first that there are two schedules of prices. The price of electricity for the building “bldg price” is rising to more than double its original price of $0.15 during the interval. The price for Electric Vehicles (EV) is fixed at the lower-than-market rate of $0.12, perhaps because public policy is set to encourage their use. Each of those price curves has an EMIX description.

In the language of EMIX and WS-Calendar, this Event contains two Resources and three Schedules. The Resources are the Electric Vehicle and the Building. The Vehicle receives one schedule of Prices. The Building receives two schedules, one dispatch based, and one price based. Both resources are located within the VEN, and any decisions about how to respond to the event are made within the VEN which is the sole point of communication for the VTN.

The duration that encompasses the event is known as the Active Period for the event. Before and after the event, there is a notification period and a recovery period, respectively. These are fixed durations communicated from the VEN to the VTN, which then must respect them in transactions it awards the VEN.

1.4.1 Streams in a DR Event

The three schedules above are conveyed using Signals which are expressed as Streams as defined above.

The dispatch level, i.e., the load reduction made by the building, varies over time. This may be tied to building capabilities, or to maintaining essential services for the occupants. It is not important to the VTN why it is constrained, only that it is.

Note that the reductions in Figure 9 do not line up with the price intervals on the bar above. In this example, the dispatch level is applied to its own WS-Calendar sequence. There is no requirement that intervals in separate streams in an event align.
An Event may be associated with Observational Streams to report back to the requester information measured or derived during the event.

### 1.4.2 The Active Period Schedule

The Active Period is a special schedule for the overall description of an Event. The Active Period may have commercial and regulatory meaning, such as a rule requiring that an Event not be longer than two hours. While an Event as described below may have many schedules as expressed in Streams, it has one Active Period.

The Active period of an event typically includes intervals in which the receiving system prepares for the event, begins its response, maintains its response, and recovers from the response. The schedules for these activities MAY be expressed using EMIX artifacts. For Power communications these can be expressed using artifacts based on EMIX Resources. The schedule for an Event MAY be expressed as can any other Sequence.

More commonly, the Active Period is expressed through a single Interval. The properties of WS-Calendar are extended in this specification to include durations to indicate the notification, ramp, and recovery periods. These are interpreted as if they are a normal sequence, constructed as indicated in Table 5.

### Table 5: Semantics of the Active Period

<table>
<thead>
<tr>
<th>Active Period elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Period</td>
<td>The nominal period of the Event. Expressed as a Vcalendar containing the Active Interval and supporting schedule information.</td>
</tr>
<tr>
<td>Active Interval</td>
<td>Interval within the Active Period whose Start Time and Duration define the period. The Active Interval may be the Designated Interval in the Sequence in the Active Period or it may be a specialized Interval as described above.</td>
</tr>
<tr>
<td>Notification Period</td>
<td>Nominally, the period expressed as a Duration between notification of the event and the commencement of the Active Interval. In distributed scenarios, a VEN may receive notification before or after this moment. Constrained devices may increase energy use during the Notification Period so as to be able to reduce energy use during the Active Interval.</td>
</tr>
<tr>
<td>Ramp Up Period</td>
<td>Period at the beginning of the Active Interval expressed as a Duration, during which a VEN moves from its former state to its requested state. If negative, then the Ramp Up occurs within the bounds of the Active Interval, i.e., it starts at the same moment as the Active Interval. If there is no Ramp Up Period, then all other rules are processed as if there were a Ramp Up Period of zero length.</td>
</tr>
<tr>
<td>Recovery Period</td>
<td>Period at the end of the Active Interval expressed as a Duration during which the effect of the response may be reversed while the system returns to its base state. For example, a system that reduces energy use during an Event by raising the air temperature may use additional energy during the recovery period while cooling the air to the normal setting. If negative, then the Recovery Period occurs within the bounds of the Active Interval, i.e., it ends at the same moment as does the Active Interval.</td>
</tr>
<tr>
<td>Tolerance</td>
<td>A collection of parameters that indicate whether there is a range of acceptable starting and ending times for the Active Period. Tolerance is used to smooth the response so that thousands of systems do not change state at the same moment.</td>
</tr>
</tbody>
</table>
2 Semantics of Energy Interoperation

As stated in Section 1, much of the core vocabulary for this specification comes from [EMIX] and [WS-Calendar]. This section introduces the remaining vocabulary for Energy Interoperation and then defines the use of that vocabulary in the higher level types.

The services of Energy Interoperation are built around exchanges of and references to these standard information artifacts.

### Table 6: Energy Interoperation Identities

<table>
<thead>
<tr>
<th>Identity Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Party</td>
<td>As described in Section 3, all interactions are between two Parties. A Party consists of a Party ID, a Party Name, and a Party Role. The Party ID is a sub-type of the UID.</td>
</tr>
<tr>
<td>Resource</td>
<td>Identifies a discrete set of capabilities that a Party may offer to a counterparty. Resources may represent specific equipment, collections of market interactions, or a detailed promise to perform. Resources are associated with a VEN during Enrollment.</td>
</tr>
<tr>
<td>Market</td>
<td>When used in this specification, a Market is a set of agreed upon assumptions and business practices. Tariffs and utility programs are examples of Markets. Each negotiation and transaction occurs within the named context of a Market.</td>
</tr>
<tr>
<td>Market Context</td>
<td>A collection of machine readable Market rules and assumptions. A Market Context is uniquely identified by a URI as defined by the EMIX Market Context. This URI can be used to retrieve the Context.</td>
</tr>
<tr>
<td>UID</td>
<td>Unique Identifier for every party, role, message, event, etc.</td>
</tr>
</tbody>
</table>

The elements above are used throughout the messages of this specification.

### 2.1 Dramatis Personae: Identifying the Actors

As described in Section Error! Reference source not found., each interaction is an interaction between two parties.

<table>
<thead>
<tr>
<th>Low Level Identity Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEN</td>
<td>As described in section Error! Reference source not found. above, A Virtual End Node is a Party acting in a specific role in a market managed by a VTN.</td>
</tr>
<tr>
<td>VTN</td>
<td>As described in section Error! Reference source not found. above, A Virtual Top Node is a Party acting in a specific role that sends events market information to a VEN.</td>
</tr>
<tr>
<td>Group</td>
<td>Resources and VENs may be the target of an Event. How group membership is identified or recognized is out of scope.</td>
</tr>
<tr>
<td>Target</td>
<td>A set of elements that collectively name which Parties should participate in an event. A Target can include Service Areas, named Groups, VENs, and Resources and other standard identifiers. The Target can be used by VENs that are also VTNs and must relay event information downstream to other VENs.</td>
</tr>
</tbody>
</table>
Figure 10: EI Target

2.1.1 Actor IDs and Roles

There is a certain fungibility of the Actor IDs in the service payloads. A Party may participate in many interactions, yet it is necessary to distinguish each Party by the role it is playing in the current interaction. Accordingly, there are named derivatives of the Actor ID for use in each situation.

Figure 11: UML Class Diagram of Party ID and its derivatives

2.2 Market Context

As defined in [EMIX], a Market Context is a URI, and it can be used to reference Standard Terms. This specification describes the expanded set of context information that is part of the EI Market Context.
The Elements of the EI Market Context are, for the most part, defined in [EMIX]. The Market Name conveys a human-readable text, perhaps for display in a user interface. As in EMIX, the Envelope contains warrants and certificates. For example, if a Market is purported to convey Green Power, however defined, that information would be conveyed in the Envelope. Two elements, Simple Levels and Application Specific Extensions bear discussion here.

2.2.1 Simple Levels

The Simple Level Context is an agreement-based interaction abstracted away from expressions of value or actual amounts. Simple Levels define levels of energy scarcity and abundance, at an agreed upon granularity. A VEN can discover Specific Levels within a Market Context.

Table 7: Simple Levels

<table>
<thead>
<tr>
<th>Level Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Level Context</td>
<td>Simple Levels are a set of simple indicators about scarcity and value, in which an ordered set of values indicate energy scarcity is above normal, normal, or below normal. Presumably, at higher levels, the VEN will use less.</td>
</tr>
<tr>
<td>Upper Limit</td>
<td>The upper level for this Context. If the Upper Limit is 5, the levels are 1-5, where 5 indicates the greatest scarcity.</td>
</tr>
<tr>
<td>Normal Value</td>
<td>The &quot;normal&quot; level indicating normal energy availability. Levels below normal indicate surplus, levels above normal indicate increasing scarcity. If the Upper Limit is 7, the levels are 1-7, and the Normal Value might be 3.</td>
</tr>
<tr>
<td>Level</td>
<td>Payload used in Signals to convey Simple Level to a VEN</td>
</tr>
</tbody>
</table>

For example, a simple program may have the levels Normal, High, and Critical. The Simple Level Context would indicate three levels with a normal value of one.

How a VEN associates particular activities and responses to the Simple Levels is out of scope for this specification.

2.2.2 Application Specific Extensions

A VTN may wish to communicate with, and a VEN may wish to allow communication with a specific Application operating within the VEN. Operating such an Application MAY be part of a specific Market Context. This specification provides explicit support for these Application Specific Extensions by means of 4 abstract types.
Application Specific Extensions

<table>
<thead>
<tr>
<th>Extensions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Specific Extension Base</td>
<td>An abstract Base Type for all other Application Specific Extensions. Application Extensions are used to provide hints to or interactions with Applications running on the other side of an interaction. They are not defined in Energy Interoperation, although there are specific conformance rules that must be followed.</td>
</tr>
<tr>
<td>Application Specific Context Base</td>
<td>An abstract class to exchange invariant or setup information with an Application running on the other side of an interaction. The Context Base is exchanged as part of a Market Context.</td>
</tr>
<tr>
<td>Application Specific Signal Base</td>
<td>An abstract class to exchange current information and varying information with an Application running on the other side of an interaction. The Signal Base is exchanged by means of an Event Signal.</td>
</tr>
<tr>
<td>Application Specific Report Base</td>
<td>An abstract class to exchange Reports with an Application running on the other side of an interaction. The Report Base is exchanged by means of an Event Report or by the Report Service.</td>
</tr>
</tbody>
</table>

The primary concern of the conformance rules for Application Specific Extensions is that they avoid redefinition of the semantics of Energy Interoperation. Prices SHALL be communicated as defines in EMIX Price Base. Schedules SHALL be communicated using the semantics of WS-Calendar. Products and things to be measured SHALL be expressed using the EMIX Item Base.

Parties wishing to exchange Application Specific Extensions SHALL extend the Signal Types and Report Types to indicate they are using their specific Payloads.

### 2.2.3 Response Smoothing

Precision of communication and response causes new problems for collections of entities and systems. With WS-Calendar and Energy Interoperation, thousands of systems and devices could respond at the same moment, causing grid instabilities or even equipment damage.

To avoid these problems, Energy Interoperation uses WS-Calendar Tolerances (Start Before, Start After, End Before, and End After) to specify a Duration in which response smoothing MAY be requested.

To further refine the expectation surrounding Smoothing, this specification defines a new Term, i.e., an extension of the EMIX Base Term, to convey expectations for smoothing the aggregate response. Because it is a Term, is can be communicated as part of a Market Context, or as part of an individual Event.

The Smoothing Term provides actionable information; of course the degree of adherence to what is an application or deployment performance characteristic is out of scope for this specification. See also Section 1.1.4.

### Table 9: Smoothing Terms

<table>
<thead>
<tr>
<th>Response Smoothing</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoothing</td>
<td>Response Smoothing defines a Term that indicates that the recipient is to ensure that the response is not in a single step. Response Smoothing is applied to the tolerance interval[s] indicated by the Start Before, Start After, End Before, and End After tolerances. The enumerated values of Smoothing are below.</td>
</tr>
<tr>
<td>Ramp</td>
<td>A smooth or uniform step ramp is indicated between the initial and end values in the respective Tolerance Interval</td>
</tr>
<tr>
<td>Uniform</td>
<td>A uniform distribution is indicated over the entire respective Tolerance Interval.</td>
</tr>
<tr>
<td>None</td>
<td>No specific smoothing is indicated. Applications need not react in a stepwise manner, so some degree of smoothing MAY occur in response to this request. If the Smoothing Term is absent, the behavior requested is the same as None.</td>
</tr>
</tbody>
</table>
2.3 Event-based Interactions

Events are stylized business interactions that are used in formal demand response environments. As described in Section Error! Reference source not found., Events are used in communications between a VTN and a VEN. An Event consists of the time periods, deadlines, and transitions during which Demand Resources perform. The VTN specifies the duration and applicability of an Event. Some deadlines, time periods, and transitions may not be applicable to all products or services.

Figure 13: Event Overview

2.3.1 The Event Descriptor

The Event descriptor contains metadata about the event itself.

<table>
<thead>
<tr>
<th>Event Descriptor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event ID</td>
<td>Identifier assigned to the Event Descriptor</td>
</tr>
<tr>
<td>Modification Number</td>
<td>If present, indicates that the event has been modified. Incremented each time the event is modified.</td>
</tr>
<tr>
<td>Modification Date and Time</td>
<td>The date and time a modification takes effect.</td>
</tr>
<tr>
<td>Modification Reason</td>
<td>Reason describing why the event is being modified. The values for reason are not specified or restricted.</td>
</tr>
<tr>
<td>Priority</td>
<td>Optional indication of the priority of an event. A given VEN or Resource may be eligible for more than one event at the same time.</td>
</tr>
<tr>
<td>Market Context</td>
<td>The overall market or program rules that govern this event.</td>
</tr>
<tr>
<td>Created Date Time</td>
<td>Indicates when this artifact was created.</td>
</tr>
</tbody>
</table>
### Event Descriptor Elements

<table>
<thead>
<tr>
<th>Event Descriptor Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Status</td>
<td>Indicates the current status of an event as of the descriptor generation. Enumerated values are: Far: Event is in the far future. The exact definition of how far in the future this refers is dependent upon the market context, but typically means the next day. Near: Event is in the near future. The exact definition of how near in the future the pending event is active is dependent on the market context. Active: Event has been initiated and is currently active. Completed: Event has completed. Cancelled: Event has been canceled. These values are similar but not identical to those used by the Event Filter as described in Section 6.2 “Special Semantics of the Event Request Operations.” The value is present in Energy Interoperation to support backward compatibility with OpenADR 1.0.</td>
</tr>
<tr>
<td>Operating Day</td>
<td>Indicates the nominal date for the event. Important for some market contexts.</td>
</tr>
<tr>
<td>Test Event</td>
<td>If present, can indicate that this event is a test event rather than an actual event.</td>
</tr>
<tr>
<td>Comment</td>
<td>Free-form information provided by the VTN</td>
</tr>
</tbody>
</table>

### 2.3.2 The Active Period

See Section 1.4 for terminology describing the periods of an event.

The Active Period is a Sequence that describes the overall schedule for an Event. The Active period is a Vcalendar type that contain a Sequence and MAY have its own properties. The Sequence of an Active Period generally falls into a common Interval pattern of Notification, Ramp-up, Active, and Recovery. The Designated Interval of the Sequence is also referred to as the Active Interval.

This stereotypic pattern can be collapsed with the Intervals for Notification, Ramp-up, and Recovery expressed as Properties of the Active Interval. Notwithstanding this common pattern, the Active Period can contain any valid Sequence, as long as the meaning conveyed is understood by both parties.

A single Event may be broadcast to many VENs with similar performance characteristics. If the VENs all perform in unison, it can create spikes (or sudden drops) in energy use that can be harmful to the distribution system. It is necessary for a VEN to be able to ameliorate this issue by requesting response smoothing as described in Section 1.1.4.

A smoothing request is indicated through the WS-Calendar Tolerance Property. This property is applied to the overall Active Period so its meaning is the same whether the simplified common pattern or a full Sequence is conveyed.
Event Signals convey the detailed information about the schedule for an event. Signals are conveyed using Streams as described in Section 1.3. When an Event conveys multiple signals, they may be aimed at different target resources in different Market Contexts, or they may use different semantics, i.e., one use Price and another use Simple Level semantics. All Event Signals have a common form.
As do all Streams, each Event Signal has a starting time, and a Tolerance (for smoothing); if absent, these are inherited from the Active Interval as if the Active Interval were a Gluon. The Time Zone is inherited from the Market Context. Each Event Signal includes a Related-To parameter to name the Designated Interval; if there is none, the first Interval is the Designated Interval. The Designated Interval has specific meaning for Sequence scheduling as defined in WS-Calendar.

2.3.3.1 Details of the Signal

Each signal includes a Market Context and optionally a Target. The Market Context and Target are used by the VEN to select which Signal, if any, to respond to. The Signal Name provides the VEN with a human-friendly description of the Signal, perhaps for display in a user interface. An EMIX Item Base enumerates what is being measured, and perhaps paid for, by the Signal. A Signal Type defines what Payload must be used throughout the signal; all Payloads in a signal MUST be of the same type. Each Interval contains a Payload, as specified by the Signal Type. An optional element, Current Value caches the current value (as of the signal creation) of the Payload.

<table>
<thead>
<tr>
<th>Table 11-: Signal Types</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Signal Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delta</td>
<td>The Payload in each Interval indicates a request to change the amount [used] by the amount in the signal as denominated by the Item Base.</td>
</tr>
<tr>
<td>Multiplier</td>
<td>The Payload in each Interval indicates a request to change the amount [used] to an amount computed by the amount in the signal times the Baseline as denominated by the Item Base.</td>
</tr>
<tr>
<td>Level</td>
<td>The Payload in each Interval indicates the Level during each Interval. See Section 2.2.1 for a description of Simple Levels.</td>
</tr>
<tr>
<td>Price</td>
<td>The Payload in each Interval indicates a price per unit as denominated by the Item Base. Price is conveyed as an EMIX Price, either a Price, a Price Multiplier, or a Price Relative. Each Payload in a Stream must contain the same type of Price. The Currency for each Price is inherited from the Market Context. In EMIX, both Price Multipliers and Prices Relatives include a Market Context; in a Payload in Signal, these are inherited from the Signal’s Market Context.</td>
</tr>
</tbody>
</table>
Signal Types | Description
---|---
Product | Signal indicates the Product for each interval. Payload Type is an EMIX Product Description.
Set-point | The Payload in each Interval indicates a requested amount (to use) as denominated by the Item Base. The amount may be more or less than the amount in the Baseline.

Parties may choose to exchange application specific payloads in signals as well. Prior to doing so, they MUST extend the Application Specific Signal Base and agree upon the Signal Type they will use. The Signal Type MUST conform to the EI Extension pattern. See Appendix C for a discussion of conforming extension.

### 2.3.4 Baselines

Baselines are streams that can incorporate signals and share many of the same elements. As some signals indicate the performance requested is relative to that in another interval, Baselines indicate the performance in that interval.

The Baseline is a signal that expresses the amount as denominated by the Item Base that is the starting point for the signal types above. The computational basis for the Baseline is not in scope for this specification. The Baseline is compared to the actual metered consumption during the Event to determine the value of the Response. Depending on the type of product or service, Baseline calculations may be performed in real time or after the fact.

Another form of the Baseline merely indicates the comparable period that is used for comparison. This enables the sender to indicate when the Baseline is drawn from without indicating the values for that Baseline period, which may not yet be known.

### 2.3.5 Opt – Making Choices

When a VEN enrolls in an event-oriented Market Context, it makes itself Available to respond to events on a given schedule. The Availability schedule may be simple (all day, all the time) or complex (weekday afternoons, on weekends with a long notice, and not on Thursday mornings during biweekly payroll). No matter how simple or complex the Availability, the VEN may choose to change it for a limited period. This decision is communicated with an Opt (as in “Opt In” and “Opt Out”).

The primary information payload for an Opt is a collection of Vavailability artifacts. An optional element inside each Availability artifact determines whether the particular repeating schedule within indicates availability or unavailability.

Business rules require that someone Opting declare their reason, using one of the specific enumerated reasons or an extension as allowed by the local Market.

<table>
<thead>
<tr>
<th>Table 12: Opt Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opt</td>
<td>Opt are used by the VEN to temporarily modify availability in the pre-existing agreement. For example, a VEN may Opt In to events during the evening, or Opt Out from events during the World Series.</td>
</tr>
<tr>
<td>Opt ID</td>
<td>A reference ID for a particular Opt notification. This identifier may be used by other entities to refer to this instance of an Opt.</td>
</tr>
<tr>
<td>Opt Type</td>
<td>Either Opt-In or Opt-Out. This element determines the processing of the Vavailability. If Opt In, then any available time is added to the pre-existing schedule. If Opt-Out, then for the period bracketed by the Availability, the schedule replaces the pre-existing schedule.</td>
</tr>
<tr>
<td>Opt Reason</td>
<td>Reason for the Opt. Enumerated reasons include: Economic, Emergency, Must Run, Not Participating, Outage Run Status, Override Status, Participating</td>
</tr>
</tbody>
</table>

The Opt Type controls specific differences in how an Opt is processed against the pre-existing availability.

**Opt-In:** After processing, the new schedule and availability is added to the existing availability for the period bounded by the Opt Availability.
Opt-Out: After processing, the new schedule and availability replace the existing availability for the period bounded by the Opt Availability.

In either case, when the bounding period is over, Availability reverts to the previous schedule.

2.4 Monitoring, Reporting, and Projection

A Party may request that another Party measure something and report back. The thing measured may include Power, Voltage, Peak, or any other attribute associated with the products exchanged. These measurements may or may not be in relation to an Event. An EiReport is the record of a measurement or series of measurements made by one Party and delivered to another.

A Party requests another Party to prepare a Report by means of a Report Request. Report Requests can be delivered using the Report service, or can accompany an Event. The Historian and Projection services also make use of the Report Request.

![Figure 16: The Report Request](image)

### Table 13: Elements of the Report Request

<table>
<thead>
<tr>
<th>Report Specifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Request ID</td>
<td>Identifies this request</td>
</tr>
<tr>
<td>Report Specifier ID</td>
<td>References the Report Specifier for this Request. The Specifier may be known from a previous request, or may be a standard Specifier within this Market Context.</td>
</tr>
<tr>
<td>Report Specifier</td>
<td>Request MAY optionally include the Report Specifier as described below.</td>
</tr>
<tr>
<td>Target</td>
<td>Standard group of Parties, Resources, Groups, etc. that the Report concerns.</td>
</tr>
<tr>
<td>Report Scheduler</td>
<td>Indication of when the report is to be run, for how long, etc.</td>
</tr>
<tr>
<td>Aggregate Report</td>
<td>As the Target of a Report Request may indicate multiple Parties or Resources, this Boolean indicates whether a single report or one for each entity matching the Target is requested.</td>
</tr>
</tbody>
</table>

2.4.1 The Report Specifier

A Party specifies what reports it wants by means of a Report Specifier. Report Specifiers may be delivered in the Report Request or be known from the Market Context.
A single Report Specifier may generate quite different Reports based upon which service it is delivered by and how it is scheduled. The elements of a Report Specifier are as follows:

**Table 14: Elements of the Report Specifier**

<table>
<thead>
<tr>
<th>Report Specifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifier ID</td>
<td>Identifies this Report Specifier</td>
</tr>
<tr>
<td>Market Context</td>
<td>The Optional Market Context MAY provide information about the Product that is being reported, or about where this Specifier came from.</td>
</tr>
<tr>
<td>granularity</td>
<td>Duration defining temporal detail, i.e., &quot;read the meter every 5 minutes.&quot;</td>
</tr>
<tr>
<td>Report Back Duration</td>
<td>Report Back to requestor, with the report-to-date at each passing of this Duration during the Report Interval. If Optional, no Report Back is expected.</td>
</tr>
<tr>
<td>Report Interval</td>
<td>Interval indicating the total span of the report. Parallel to Active Interval. May be influenced by a Gluon in the Report Scheduler. If the Interval contains a Start Date and no Duration, then the Report is to begin at the Start date and continue indefinitely.</td>
</tr>
<tr>
<td>Specifier Payload</td>
<td>The Specifier Payload indicates exactly what is to be in the report.</td>
</tr>
</tbody>
</table>

2.4.1.1 The Report Specifier Payload

The Specifier Payload indicates exactly what is in the Report. It consists of an [EMIX] ItemBase and a Report Type.

**Table 15: Report Specifier Payload**

<table>
<thead>
<tr>
<th>Report Specifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rID</td>
<td>Identifies this Payload. If only one Payload is requested, the rID should be omitted; if multiple Payloads are requested in the same Report, each should have an rID.</td>
</tr>
<tr>
<td>Item Base</td>
<td>The Item Base is the core of an EMIX Product Description. Examples of Item Base denominated value include Real Power, Real Energy, Voltage, et al.</td>
</tr>
<tr>
<td>Report Type</td>
<td>Defines what is being measured and reported. Measurements are in units of Item Base unless the Report Type indicates otherwise.</td>
</tr>
</tbody>
</table>

The Report Type specifies what is measured and, sometimes, how it is measured.
2.4.1.2 The Report Types

Report Types are an enumeration that indicates how the Item Base is to be measured. These enumerations parallel the Signal Types used in Events.

| Report Types     | Description
|------------------|--------------------------------------------------------
| Reading          | Report indicates a Reading, as from a meter. Readings are moments in time; changes over time can be computed from the difference between successive readings. Payload Type is Float.
| Usage            | Report indicates an amount of units (denominated in Item Base or in the EMIX Product) over a period. Payload Type is Quantity. A typical Item Base is Real Energy.
| Demand           | Report indicates an amount of units (denominated in Item Base or in the EMIX Product). Payload Type is Quantity. A typical Item Base is Real Power.
| Set Point        | Report indicates the amount (denominated in Item Base or in the EMIX Product) currently set. May be a confirmation/return of the set point control value sent from the VTN. Payload Type is Quantity. A typical Item Base is Real Power.
| Delta Usage      | Change in Usage as compared to the Baseline
| Delta Set point  | Changes in Set point from previous schedule
| Delta Demand     | Change in Demand as compared to the Baseline
| Baseline         | Can be Demand or Usage, as indicated by ItemBase. Indicates what the amount would be if not for the Event or Regulation. Report is of the format Baseline.
| Deviation        | Difference between some instruction and actual state.
| Average Usage    | Average usage over the duration indicated by the Granularity
| Average Demand   | Average usage over the duration indicated by the Granularity
| Operating State  | Generalized state of a resource such as on/off, occupancy of building, etc. No ItemBase is relevant. Requires an Application Specific Payload Extension.
| Up Regulation Capacity Available | Up Regulation capacity available for dispatch, expressed in EMIX Real Power. Payload is always expressed as positive Quantity.
| Down Regulation Capacity Available | Down Regulation capacity available for dispatch, expressed in EMIX Real Power. Payload is always expressed as positive Quantity.
| Regulation Set point | Regulation set point as instructed as part of regulation services
| Current Storage  | Item Base is expressed as Real Energy and Payload is expressed as a Quantity.
| Target Storage   | Item Base is expressed as Real Energy and Payload is expressed as a Quantity.
| Available Storage Capacity | Capacity available for further energy storage, presumably to get to Target Storage.
| Price            | Report Prices per ItemBase at each interval
| Level            | Report Simple Level at each interval. ItemBase is not meaningful.

Report Type is implemented as an enumerated string with extensibility. Parties wishing to extend the enumeration MUST defined the report payload requirements.
2.4.2 Report Scheduler

The report scheduler is an abstract type that specifies how often and for how long a report will be prepared. The Report Scheduler adds flexibility and consistency by enabling a single Report Specifier to be used in multiple scenarios. One option for Report Scheduler enables a Report Request to be associated with an Event.

Table 17: Types of Report Scheduler

<table>
<thead>
<tr>
<th>Report Scheduler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Gluon</td>
<td>Associates a Report Request with a particular event. This type consists of a Gluon and a reference to the Event ID. The Gluon sets the Report Interval relative to the Active Interval of the Event. For example: SS -T20M. The Report interval starts 20 minutes before (-T20M) the Active Interval starts (Start to Start). FF T1H. The Report interval Finishes 1 hour after (T1H) the Active Interval Finishes (Finish to Finish). If absent, the Report Interval is the same as the Active Interval, i.e., the Report runs during Active Interval. The Event ID indicates the Event this report is related to. If absent, the Report Request must be delivered as part of an EiEvent.</td>
</tr>
<tr>
<td>Request Report Gluon</td>
<td>Used if the Report Specifier includes a Report Interval to influence the expression of that Interval. Information in the Gluon is inherited by the Report Interval in conformance with WS-Calendar.</td>
</tr>
<tr>
<td>Request Report Interval</td>
<td>The Interval in Scheduler is the Report Interval for the Report. If the Specifier included an Interval, it is replaced by the one in the Schedule.</td>
</tr>
<tr>
<td>Request Report Snap</td>
<td>Indicates that the readings indicated by the Specifier are to be made once at the Status Date and Time and then returned to the Requester. If the Status Date and Time are omitted, then the Snap is to be made at the time of receipt.</td>
</tr>
</tbody>
</table>
2.4.2.1 UML Diagram of Report Scheduler

Figure 19: UML Diagram of Report Scheduler
2.5 Reports, Snaps, and Projections

Reports are simple Streams with some metadata identifying the report and a collection of Intervals containing the Payloads for each [measurement]. Reports can be of the past, the present, or the future. A Report appears as a series of [measurements] in the past. A Snap is a Report made as of a single moment. A Projection is in the same form as a report, but it includes projections of what will be in the future, including a confidence level in the payload.
Table 18: Reports

<table>
<thead>
<tr>
<th>Report Metadata</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Request ID</td>
<td>Identifies the Request that resulted in this Report.</td>
</tr>
<tr>
<td>Report Specifier ID</td>
<td>Identifies the Report Specifier that resulted in this Report.</td>
</tr>
</tbody>
</table>

The above information is sufficient to uniquely identify each Report, why it was made, and to what specifications. The full form of a report is as follows in Figure 21.

Table 19: Elements of Reports

<table>
<thead>
<tr>
<th>Report Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date and Time</td>
<td>Indicates the beginning of the Report</td>
</tr>
<tr>
<td>Duration</td>
<td>Indicates the Duration of each Interval in the Report</td>
</tr>
<tr>
<td>Related To</td>
<td>Inherited from Stream Base but not used in Reports. Must be Ignored.</td>
</tr>
<tr>
<td>Report Name</td>
<td>Optional human-friendly name for the report</td>
</tr>
</tbody>
</table>
### Report Description

<table>
<thead>
<tr>
<th>Report Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Description</td>
<td>Type describing the make-up of the report which MAY not be entirely determinable from the Specifier. Also, explains the interpretation of each Value.</td>
</tr>
<tr>
<td>Created Date and Time</td>
<td>Indicates when the Report was prepared for delivery to the requestor.</td>
</tr>
</tbody>
</table>

#### 2.5.2 Report Description

The Report Description indicates what is in the Report, which may be different from what was specified, particularly if multiple elements were in the Target. A Report may include multiple Report Descriptions if multiple payloads are delivered in each interval. Conversely, if the Recipient is able to rely completely on the Report Specifier, the Report Description MAY be omitted.

The Elements of the Report Description are as follows:
### Table 20: Elements of the Report Description

<table>
<thead>
<tr>
<th>Report Elements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rID</td>
<td>Optional report identifier required only if multiple payloads are delivered in each Interval.</td>
</tr>
<tr>
<td>Subject</td>
<td>Identifies the specific thing or things being measured in this report. Subject is in the form of a Target, which means it can include one or more Parties, Resources, Assets, Groups, etc.</td>
</tr>
<tr>
<td>Data Source</td>
<td>Identifies the Source of the information or measurement provided. A common use is to identify the MRIDs of the meter[s] that apply to the Subject. Data Source is in the form of a Target.</td>
</tr>
<tr>
<td>Report Type</td>
<td>Identifies what is the meaning of each measurement, as defined in Section 2.4.1.2.</td>
</tr>
<tr>
<td>Item Base</td>
<td>Identifies the Units being measured, unless the Report Type indicates this element is meaningless.</td>
</tr>
<tr>
<td>Reading Type</td>
<td>If present, indicates metadata about the Readings, i.e., direct measurement or computation. Conforming profiles MAY ignore Reading Type.</td>
</tr>
<tr>
<td>Aggregate Report</td>
<td>Identifies whether each payload represents an individual subject, or the sum of multiple subjects.</td>
</tr>
</tbody>
</table>

#### 2.5.3 Report Payloads

The details in each Interval in a Report bear a lot of similarity to those in the Signals. In many cases, a Signal requests that a system provide something similar to its Signal Value. Reporting back in the same format enables ready comparisons. These values are conveyed in the Payload.

Signals, though, are ideal. Reports describe real world effects, and therefore messy. For this reason, Report Payloads include some additional information.

Figure 23: the Report Payload

Figure 23 shows the information qualifications alongside the Payload. If an Application within a VEN has specific reporting requirements, a new Payload Type can be derived from the abstract Payload Application Specific type; a type so derived can be delivered by a conforming report service.
### Table 21: Report Payload Qualifiers

<table>
<thead>
<tr>
<th>Metadata</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence</td>
<td>An optional information structure that indicates in each interval how likely the information is to be precise.</td>
</tr>
<tr>
<td>Reading Type</td>
<td>An enumerated indication of different ways to derive a reading</td>
</tr>
<tr>
<td>Accuracy</td>
<td>An indicator of Payload accuracy</td>
</tr>
</tbody>
</table>

#### 2.5.3.1 Reading Types

The Reading Type describes the information returned in a report. Specifically, the Reading Type describes how the number in the payload was arrived at. The Reading Type MAY be in the stream Gluon, and be inherited by each Interval in the Sequence (or by the Snap, if present). The Reading Type MAY also appear in any Interval where the reporting system is indicating that one payload differs from others in the Sequence. Reading Types are described in Table 22.

<table>
<thead>
<tr>
<th>Reading Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Read</td>
<td>Reading is read from a device that increases monotonically, and usage must be computed from pairs of start and stop readings.</td>
</tr>
<tr>
<td>Net</td>
<td>Meter or [resource] prepares its own calculation of total use over time</td>
</tr>
<tr>
<td>Allocated</td>
<td>Meter covers several [resources] and usage is inferred through some sort of pro rata computation.</td>
</tr>
<tr>
<td>Estimated</td>
<td>Used when a reading is absent in a series in which most readings are present.</td>
</tr>
<tr>
<td>Summed</td>
<td>Several meters together provide the reading for this [resource]. This is specifically a different than aggregated, which refers to multiple [resources] in the same payload. See also Hybrid.</td>
</tr>
<tr>
<td>Derived</td>
<td>Usage is inferred through knowledge of run-time, normal operation, etc.</td>
</tr>
<tr>
<td>Mean</td>
<td>Reading is the mean value over the period indicated in Granularity.</td>
</tr>
<tr>
<td>Peak</td>
<td>Reading is Peak (highest) value over the period indicated in granularity. For some measurements, it may make more sense as the lowest value. May not be consistent with aggregate readings. Only valid for flow-rate Item Bases, i.e., Power not Energy.</td>
</tr>
<tr>
<td>Hybrid</td>
<td>If aggregated, refers to different reading types in the aggregate number.</td>
</tr>
<tr>
<td>Contract</td>
<td>Indicates reading is pro forma, i.e., is reported at agreed upon rates</td>
</tr>
<tr>
<td>Projected</td>
<td>Indicates reading is in the future, and has not yet been measured.</td>
</tr>
</tbody>
</table>

#### 2.5.3.2 Contrasting semantics of Summary and Aggregate in Reports

Consider the following industrial facility with a single ESI acting as a VEN. This facility chose to offer four Resources to its VTN: one industrial Resource and three office Resources, one for each floor. Two of the office Resources, Floor 2 and Floor 3, have their own zones and meters. Floor 1 has two zones, 1A and 1B, that are metered separately. The three office Resources are all in a single Group, Office. The single industrial Resource is in its own Group, Factory.
A Usage report with a Target of Office applies to three Resources, Floor 1, Floor 2, and Floor 3. If the Aggregate flag is True, the VEN prepares a single report that aggregates the information from all three Resources. If a report Target indicates Industrial or Factory, Group or Resource, there is no distinction between an Aggregate or non-Aggregate request.

The Data Sources for the Usage Reports are the Meters, M1-M5. The Report for Floor 3 has a Data Source of M5. The Report for Floor 2 has a Data Source of M4. The Report for Floor 1 has two data sources, M2 and M3, and the single Reading for Floor 1 is of the Type “Summary”. 

Aggregate refers to the combining of multiple Subjects (things named in Target) into a single report; Summary refers to the combination of multiple Data Sources [meters] into a single value.
2.5.4 UML Diagram of Report

Figure 25: UML Class Diagram of Reports

2.6 Responses and Error Reporting

All Services share a common Response. The Response shares a common extensible code, a readable description, and a reference to the Message that this is in response to.

<table>
<thead>
<tr>
<th>Table 23: Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Response Elements</strong></td>
</tr>
<tr>
<td>EI Response</td>
</tr>
</tbody>
</table>
Response Elements | Description
--- | ---
Response Code | Code consisting of 3 digits for automated processing. The simplest devices need understand only the first digit, others are for extension as needed within the higher order error indicated by the first digit. 1xx: Informational - Request received, continuing process 2xx: Success - The Request was successfully received, understood, and accepted 3xx: Pending - Further action must be taken in order to complete the Request 4xx: Requester Error - The request contains bad syntax or cannot be fulfilled 5xx: Responder Error - The responder failed to fulfill an apparently valid request xx is used for defining more fine grained errors. Where possible, the HTTP errors should be used.
Response Description | Optional String describing the response or the reason for the response
Message UID | Reference to the Message that elicited this response
Response Terms Violated | Optional Array of EMIX Terms and Response Descriptions to provide a machine interpretable Response. For example, if the Request fails because it violated the "Minimum Notification Duration" of one hour, the responder could send back the Term (with value) and an Response Description.

2.6.1 Event Responses

Responses to events are not stateless, so they require further information. All Responses regarding Events have the elements in Table 24 in addition to the elements listed in Table 23.

| Event Responses | Description |
|--- | ---
| Event ID | ID of the Event which caused this Response |
| Modification Number | Modification Number of the Message about an Event that caused this Response |
| Opt Type | Indicates whether this Response results in a VEN Opting In or Opting Out of the Event. |

Some services communicate multiple messages, and the different messages may warrant different responses. In these cases, there is a single EiResponse (or EiEventResponse) which conveys an overall response. If this overall response is Success (2xx), then there is no need for the recipient to examine the message further. If the overall Response is anything other than success, then the response for each Element in the original Request can be found by examining the array of responses (type responses) or the array of Event Responses (type eventResponses) for detailed information.

2.6.2 References in Responses

Response is a general Type that must reference any number of messages, reports, requests, etc. These critical cross interaction types are each identified by a Reference ID. The Reference ID for each is derived from a common refID type that enables type-safe substitution in Response and in other payloads.

---

- 34 - IEC stage(CDV,FDIS...) 6XXXX © IEC:201X
### 2.7 Availability Behavior

In different Market Contexts, Availability is interpreted differently by the VTN. This availability behavior is published as part of the EI Market Context as it is in effect a meta-term for the market.

#### Table 25: Availability Behavior

<table>
<thead>
<tr>
<th>Availability Behavior</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavior</td>
<td>When an Event is issued by the VTN, it is validated against the parameters and constraints that were established when the Market Context was set up, i.e., the market Rules support Events between 12:00 and 16:00. If the Event is not within 12:00 and 16:00 then VEN must take some action to resolve the conflict.</td>
</tr>
<tr>
<td>Accept</td>
<td>Simply accept the issued DR event regardless of any conflicts</td>
</tr>
<tr>
<td>Reject</td>
<td>Reject any DR events that conflict with configured Availability</td>
</tr>
<tr>
<td>Restrict</td>
<td>Modify the DR event parameters so that they legally fall within the bounds of the configured parameters.</td>
</tr>
</tbody>
</table>
Introduction to Services and Operations

In the following sections services and operations consistent with [SOA-RM] are described. For each service operation there is an actor that invokes the service operation and one that provides the service. These roles are indicated by the table headings Service Consumer for the actor or role that consumes or invokes the service operation named in the Operation column and Service Provider for the actor or role that provides or implements the service operation as named in the Operation column.

This terminology is used through all service definitions presented in this specification.

The column labeled Response Operation lists the name of the service operation invoked as a response. Most operations have a response, excepting primarily those operations that broadcast messages. The roles of Service Consumer and Service Provider are reversed for the Response Operation.

All communication between customer devices and energy service providers is through the ESI.

For transactive services any party may receive tenders (priced offers) of service and possibly make tenders (priced offers) of service.

Any party using Transactive Energy services may own generation or distributed generation or reduce or increase energy from previously transacted energy amounts. These activities are not identified in transactive services. The dispatch of these resources and the use of energy by a party are influenced by tenders between Parties that may result in new Transactions and changes in operations.

The VEN/VTN services provide a characterization of the aggregate resources of a VEN that may be communicated to the VTN; that relationship depends also on the EiMarketContext in which the interactions take place.

The next section describes the role of Resources, Curtailment and Generation. In a transactive approach tendering and prices are used by parties to discover and negotiate transactions that respect the preferences of each party and energy usage, generation, storage and controllability directly available to each party. There is no formal communication of resource characteristics in the transactive approach.

3.1 Resources, Curtailment, and Generation

If the VEN participates in a demand response program or provides distributed energy resources, its ESI is the interface to at least one dispatchable resource (Resource), that is, to a single logical entity. A Resource may or may not expose any fine structure. The Resource terminology and the duality of generation and curtailment are from [EMIX].

Under a demand response program, a Resource is capable of shedding load in response to Demand Response Events, Electricity Price Signals or other system events (e.g. detection of under-frequency). The VTN can query the actual state of a Resource with the EiReport service and request ongoing information. The VEN can query the status of the VTN-VEN relationship using the EiRequestEvent operation.

Alternatively, a Resource may provide generation in response to similar information. The net effect is the same.

3.2 Structure of Energy Interoperation Services and Operations

Energy Interoperation defines a web services implementation to formally describe the services and interactions although fully compliant services and operations may be implemented using other technologies.

The services presented in this specification are divided into five broad categories:

1 A finer level of granularity is sometimes called an asset. Assets are not in scope for this specification.
Transactive Services—for implementing energy transactions, registration, and tenders.

Event Services—for implementing events and linked Reports

Report Services—for exchanging remote sensing and feedback.

Enrollment Services—for identifying and qualifying service providers, resources, and more

Support Services—for additional capabilities

The structure of each section is a table with the service name, operations, service provider and consumer, and notes in columns.

The services are grouped so that profiles can be defined for purposes such as price distribution, and Demand Response (with the functionality of [OpenADR]). This specification defines three profiles, the OpenADR Profile, the TeMIX (Transactive EMIX) Profile, and the Price Distribution Profile.

The normative XML schemas are in separate files, accessible through the [namespace] on the cover page.

### 3.3 Naming of Services and Operations

The naming of services and operations follows a pattern. Services are named starting with the letters Ei capitalization which follows the Upper Camel Case convention. Operations in each service use one or more of the following patterns. The first listed is a fragment of the name of the initial service operation; the second is a fragment of the name of the response message which acknowledges receipt, describes errors, and may pass information back to the invoker of the first operation.

- **Create**—**Created** An object is created and sent to the other Party
- **Cancel**—**Canceled** A previously created request is canceled
- **Request**—**Reply** A request is made for all objects of the specified type previously created and relevant to this VTN-VEN relationship
- **Distribute** An object (such as a price quote, a curtailment or generation request) is created and sent without expectation of response.

For example, to construct an operation name for the EiEvent service, “Ei” is concatenated with the name fragment (verb) as listed. For example, an operation to cancel an outstanding operation or event is called **EiCancelEvent**.

The pattern of naming is consistent with current work in the IEC Technical Committee 57 groups responsible for the [TC57CIM].

### 3.4 Push and Pull Patterns

The Service Operation naming includes application-level acknowledgements, which in nearly every case carry application-level information, and allow for both push and pull of messages. This description applies to both transactive and VTN/VEN interactions as both are performed by Parties taking on various roles.

Both Push and Pull are with respect to the invoker of the operation. So if a Party produces information that describes a price quote, it can invoke (in the case of Push) an operation to send it to one or more other Parties. In the alternative, each Party (in the case of Pull) can invoke a request for information by polling, or pulling it from another Party.

The Pull operation is performed by the Party invoking the Request service operation pattern and fulfilled with a Reply service operation pattern invoked by the receiving Party.
So a series of Push operations from one Party to a counter-Party is analogous to a series of Pull operations from the counter-Party to the Party.

In the VTN-VEN context, a series of Push operations from a VTN to its VENs is analogous to a series of Pull operations from the VEN to its VTN; by examining (e.g.) the absence of an Event that was visible on a previous Pull the VEN can infer that that Event was canceled. The VEN could then send a Canceled service operation as if it had received a Cancel service operation.

One special case is the Distribute pattern, which expects no response to the invoker.

The service quality of the Pull operations (and in particular the load on the VTN from repeated polling) is not in scope for this specification.

3.5 WSDL Integration

A WSDL represents a contract between two systems that are being integrated. As such additional attributes may need to be passed in addition to the attributes that are specific to a message payload (representing the core set off information being passed). At a high level, any given integration may need to include a header, request, and/or reply in addition to the message payload as shown in the figure below.

![Figure 27: Generalized view of the high-level message structure](image)

For example, for WSDL-based integration, details regarding the specifics of a demand response event are contained in the message payload. However, additional details that work to ensure the successful integration may be included in the header, request, or reply.

A message header contains information about the sender and receiver of the message or other information used to correlate the service request, to guarantee delivery, or to support non-repudiation as seen in the [non-normative] figure below.

Message headers are out of scope for this specification.

3.6 Description of the Services and Operations

Each service is described as follows. In the sections that follow, we will:

- Describe the service
- Show the table of operations
- Show the interaction patterns for the service operations in graphic form
- Describe the information model using [UML] for key artifacts used by the service
- Describe the operation payloads using [UML] for each operation

3.7 Responses

In a service interaction, responses may need to be tracked to determine if the transaction is successful or not. This may be complicated by the fact that any given transaction may involve the transmission of one or more information objects.

The class diagram below reflects the generic response.
Figure 28: Example of generic error response for a service operation

The Reference ID (refID) identifies the artifact or message element that this response is to. The response code indicates success or failure of the operation requested. The Response Description is unconstrained text, perhaps for use in a user interface.

There is no exhaustive list of all possible Response Codes. The Response Codes are intended to enable even the smallest device to interpret Response. This specification uses a pattern consisting of a 3 digit code, with the most significant digit sufficient to interpret success or failure. This pattern is intended to support that smallest device, while still supporting more nuanced messages that may be developed.

- 1xx: Informational - Request received, continuing process
- 2xx: Success - The action was successfully received, understood, and accepted
- 3xx: Pending - Further action must be taken in order to complete the request
- 4xx: Requester Error - The request contains bad syntax or cannot be fulfilled
- 5xx: Responder Error - The responder failed to fulfill an apparently valid request

While the only value of xx that is defined as of this version is 00, conforming specifications may extend these errors to defining more fine grained errors. These errors should extend the pattern above, though. A response code such as 403 should always be within the realm of Requester Error.

3.7.1 Terms Violated

Terms Violated is an optional element of a Response. Terms communicate business expectations. It may be that a Service Request fails not because it is improperly formed, but because it violates one or more of these business rules. For example, a Market Term may indicate a 20 minute notification duration. A Service Request that asks for a performance with only a 5 minute notification violates that Term. By passing that Term back in the Response, that service provider can make known what its requirements are.

It is outside the scope of this specification whether a provider MAY present terms while still accepting a Service.

3.7.2 Response Derivations

Because some responses require additional context relative to the Service requested, the same types derive from and extend the Response type.
1.7.2.1 Event Responses

Event Responses are derived from the Response Type and add elements useful for Event-based interactions. Event Responses include Event ID and Modification Number to indicate exactly which Event they are responding to. Event Responses also include the Opt Type (Opt In or Opt Out) to describe what response is being made to an event.

1.7.2.2 Enrollment Responses

Enrollment Responses are derived from the Response Type and add elements useful for Event-based interactions. The Enrollment response includes an Enrollment ID to indicate which Enrollment is being referenced.

Enrollment establishes a business relationship between a Party and a particular Market Context. A Party may be enrolled in several Market Contexts. Enrollment Responses include the Market Context that is affected by the Response.

A single request to Enroll may create many Enrollment IDs. For example, a Party offering several Resources may get an Enrollment ID for each. Similarly, a single Resource may become enrolled in both a power and a regulation Market Context. An Enrollment Response includes a Market Context to indicate which Market Context was affected.

As stated above, a single request to Enroll may create many Enrollment IDs. It can be helpful to know the original request’s reference ID to understand the Response. An Enrollment Response MAY include an Original Reference ID.

1.7.3 Compound Responses

Many service interactions may affect a number of messages. For example, a single service interaction may include multiple Tenders, or Events. A single Enrollment request may result in multiple Enrollments. All such Responses have the pattern of a single Response (or Event Response, or Enrollment Response) accompanied by a collection of Responses. This specification defines the collections of Responses, Event Responses, and Enrollment Responses.

The end-point receiving a compound Service Payload, including both single Responses and collections of Responses follows these rules:

- If the Response indicates success, there is no need to examine each element in the Responses.
- If some elements fail and other succeed, the Response will indicate the error, and the recipient should evaluate each element in the Responses to discover which components of the operation failed.
### Summary of Response and Responses

A Response returns the success or failure of the entire operation. The Responses returns an ID and a Response for each.

It is MANDATORY to return errors in responses. It is OPTIONAL to return successes in responses. For Cancel, in particular, it is not mandatory to return any responses if the entire operation was completed successfully. The pattern is to return those that have failed (required) and those that succeeded (optional).

### Requests

Each of the Services includes a Request, which is essentially a status update. Consider the Service Foo. A Request means "tell me all the Foos that we have outstanding." The meaning of outstanding varies from Service to Service. In general, either party may make invoke the Request Service on the other. Tell me all the Quotes you have given me is the mirror of Tell me all the Quotes you have received from me. Each Request shares the same semantics.

Each optional element in a Request refines or narrows the scope of the Request by narrowing the request to only those Foos for which the named elements match. If there are more than one instance of the same named element, then this restriction element is treated as if a logical OR were applied, i.e., element = A OR element = B. Where more than one type of element is named, then the restriction is treated as an AND, i.e., element A = "foo" AND element B = "fie".

A special element that is included in most Requests is the Interval. The Interval is treated as a temporal restriction. For example, an Interval that encompasses a business day can request all Foo for delivery on that day. Intervals MAY be open-ended. An Interval conveying only a Start Date matches all Foo that are current from that date and time forward. An Interval conveying only an End Date matches all Foo that are current at that date and time. If there is any ambiguity
about what “matches” means, it is defined within the Service section below, c.f., the definition of pending Events in Section 6.2 “Special Semantics of the Event Request Operations”.
24 Transactive Services

Transactive Services define and support the lifecycle of transactions inside an overarching agreement, from initial quotations and indications of interest to final settlement. The phases are:

- **Registration**—to enable further phases.
- **Pre-Transaction**—non-binding quotes and binding tenders for transactions.
- **Transaction Services**—execution and management of transactions including transaction with optionality.
- **Post-Transaction**—settlement, energy used or demanded, payment, position.

For transactive services, the roles are **Parties** and **Counterparties**. For event and resource services, the Parties adopt a VTN or VEN role for interactions. The terminology of this section is that of business agreements: tenders, quotes, and transaction execution and (possibly delayed) performance under an optional or DR transaction.

The register services identify the parties for future interactions. This is not the same as (e.g.) a program registration in a demand response context—here, registration can lead to exchange of tenders and quotes, which in turn may lead to a transaction which will determine the VTN and VEN roles of the respective parties.

### 4.1 EiRegisterParty Service

The EiRegisterParty service operations create a registration for potential Parties in interactions. This is necessary in advance of an actor interacting with other parties in various roles such as VEN, VTN, tenderer, and so forth.

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiRegisterParty</td>
<td>EiCreateParty Registration</td>
<td>EiCreatedParty Registration</td>
<td>Party</td>
<td>Party</td>
<td>Create and send a Party Registration request</td>
</tr>
<tr>
<td>EiRegisterParty</td>
<td>EiRequestParty Registration</td>
<td>EiReplyParty Registration</td>
<td>Party</td>
<td>Party</td>
<td>Request semantics with optional Interval</td>
</tr>
<tr>
<td>EiRegisterParty</td>
<td>EiCancelParty Registration</td>
<td>EiCanceledParty Registration</td>
<td>Party</td>
<td>Party</td>
<td>Cancel one or more Party Registrations</td>
</tr>
</tbody>
</table>

Formatted: No bullets or numbering
4.1 Interaction Pattern for the EiRegisterParty Service

This is the [UML] interaction diagram for the EiRegisterParty Service.

![Interaction Diagram for EiRegisterParty Service](image)

**Figure 30**: Interaction Diagram for EiRegisterParty Service

4.1.2 Information Model for the EiRegisterParty Service

The details of a Party are outside the scope of this specification. The application implementation needs to identify additional information beyond that in the class EiParty.

```xml
<xsd:complexType name="EIPartyType">
  <xsd:sequence>
    <xsd:element name="partyID" type="xsd:integer" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="partyName" type="xsd:string" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="partyRole" type="xsd:string" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

**Figure 31**: EIParty UML Class Diagram
### 9.1.3 Operation Payloads for the EiRegisterParty Service

The [UML] class diagram describes the payloads for the EiRegisterParty service operations.

#### XSD Complex Type

- **EiCreatePartyRegistrationType**
  - `eRegistrationId`: EiRegistrationInfoType
  - `reHistoricID`: intID

- **EiCreateReplyPartyRegistrationType**
  - `eResponse`: EiResponseInfo
  - `reHistoricID`: intID

- **EiCreateReplyRequestPartyRegistrationType**
  - `externalPersonnelInCalendarInterval`: [0..1]
  - `reHistoricID`: intID

- **EiCreateCancelPartyRegistrationType**
  - `reRegistration`: EiRegistrationInfoType
  - `reHistoricID`: intID

- **EiCreateCancelReplyPartyRegistrationType**
  - `eResponse`: EiResponseInfo
  - `reHistoricID`: intID

- **EiRegisterRequestContextType**
  - `marketContext`: EiMarketContextType
  - `reHistoricID`: intID

---

**Figure 32:** UML Class Diagram for EiRegisterParty Service Operation Payloads
4.2 Pre-Transaction Services

Pre-transaction services are those between parties that may or may not prepare for a transaction. The services are EiTender and EiQuote. A quotation is not a tender, but rather a market price or possible price, which needs a tender and acceptance to reach a transaction.

Price distribution, which is sometimes referred to as price signals, is accomplished using the EiQuote and EiTender services. Quotes are indications of a possible tender price; they are not actionable. A Tender offers prices at which Transactions may be made; they are actionable.

As with other services, a Party MAY inquire from a counterparty what offers the counterparty acknowledges as open by invoking the EiSendTender service to receive the outstanding tenders.

There is no operation to “delete” a quote; when a quote has been canceled the counterparty MAY delete it at any time. To protect against recycled or dangling references, the counterparty SHOULD invalidate any identifier it maintains for the cancelled quote.

Tenders, quotes, and transactions are [EMIX] artifacts, which contain terms such as schedules and prices in varying degrees of specificity or concreteness.

Table 27: Pre-Transaction Tender Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiTender</td>
<td>EiCreateTender</td>
<td>EiCreatedTender</td>
<td>Party</td>
<td>Party</td>
<td>Create and send Tender</td>
</tr>
<tr>
<td>EiTender</td>
<td>EiRequestTender</td>
<td>EiReplyTender</td>
<td>Party</td>
<td>Party</td>
<td>Request outstanding Tenders; request semantics with optional time Interval</td>
</tr>
<tr>
<td>EiTender</td>
<td>EiCancelTender</td>
<td>EiCanceledTender</td>
<td>Party</td>
<td>Party</td>
<td>Cancel one or more Tenders</td>
</tr>
<tr>
<td>EiTender</td>
<td>EiDistributeTender</td>
<td>--</td>
<td>Party</td>
<td>Party</td>
<td>For broadcast or distribution of Tenders</td>
</tr>
</tbody>
</table>

Table 28: Pre-Transaction Quote Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiQuote</td>
<td>EiCreateQuote</td>
<td>EiCreatedQuote</td>
<td>Party</td>
<td>Party</td>
<td>Create and send a quote</td>
</tr>
<tr>
<td>EiQuote</td>
<td>EiRequestQuote</td>
<td>EiReplyQuote</td>
<td>Party</td>
<td>Party</td>
<td>Request outstanding Tenders; request semantics with optional time Interval</td>
</tr>
<tr>
<td>EiQuote</td>
<td>EiCancelQuote</td>
<td>EiCanceledQuote</td>
<td>Party</td>
<td>Party</td>
<td>Cancel one or more quotes</td>
</tr>
<tr>
<td>EiQuote</td>
<td>EiDistributeQuote</td>
<td>--</td>
<td>Party</td>
<td>EiTarget</td>
<td>For broadcast or distribution of quotes</td>
</tr>
</tbody>
</table>

Formatted: No bullets or numbering
4.2.1 Interaction Pattern for the EiTender and EiQuote Services

This is the [UML] interaction diagram for the EiTender Service.

Figure 33: Interaction Diagram for the EiTender Service
This is the [UML] interaction diagram for the EiQuote Service

**Figure 34**: Interaction Diagram for the EiQuote Service

### 4.2.2 Information Model for the EiTender and EiQuote Services

The information model for the EiTender Service and the EiQuote Service artifacts is that of [EMIX]. EMIX provides a product description as well as a schedule over time of prices and quantities.
4.2.3 Operation Payloads for the EiTender Service

The [UML] class diagram describes the payloads for the EiTender and EiQuote service operations.

Figure 35: UML Class Diagram for the Operation Payloads for the EiTender Service
## 4.3 Transaction Management Services

The service operations in this section manage the exchange of transactions. For example, in demand response, the overarching agreement is the context in which events and response take place—what is often called a *program*. This agreement is identified by the information element Market Context here and elsewhere.
There is no EiCancelTransaction or EiChangeTransaction operations. As in distributed agreement protocols, a compensating transaction SHOULD be created as needed to compensate for any effects.\(^2\)

### Table 29: Transaction Management Service

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiTransaction</td>
<td>EiCreateTransaction</td>
<td>EiCreatedTransaction</td>
<td>Party</td>
<td>Party</td>
<td>Create and send Transaction</td>
</tr>
<tr>
<td>EiTransaction</td>
<td>EiRequestTransaction</td>
<td>EiReplyTransaction</td>
<td>Party</td>
<td>Party</td>
<td>Request extant Transactions</td>
</tr>
</tbody>
</table>

#### 4.3.1 Interaction Patterns for the EiTransaction Service

This is the [UML](https://www.uml.org) interaction diagram for the EiTransaction Service:

![Interaction Diagram for the EiTransaction Service](image)

#### 4.3.2 Information Model for the EiTransaction Service

Transactions are [EMIX](https://www.emix.org) artifacts with the identification of the Parties.

---

\(^2\) This is consistent with the way that distributed agreement protocols such as [WS-BusinessActivity](https://www.w3.org) manage compensation rather than cancelation.
### 4.3.3 Operation Payloads for the EiTransaction Service

The [UML] class diagram describes the payloads for the EiTransaction service operations.

```xml
<xsd:complexType name="EiCreateTransactionType">
  <xsd:element name="counterPartyID" type="xsd:string"/>
  <xsd:element name="eiTransaction" type="EiTransactionType[0..1]"/>
  <xsd:element name="partyID" type="xsd:string"/>
  <xsd:element name="requestID" type="xsd:string"/>
</xsd:complexType>

<xsd:complexType name="EiCreatedTransactionType">
  <xsd:element name="counterPartyID" type="xsd:string"/>
  <xsd:element name="eiResponse" type="EiResponseType"/>
  <xsd:element name="partyID" type="xsd:string"/>
  <xsd:element name="responses" type="ArrayOfResponses[0..1]"/>
  <xsd:element name="transactionID" type="xsd:string"/>
</xsd:complexType>

<xsd:complexType name="EiRequestTransactionType">
  <xsd:element name="counterPartyID" type="xsd:string[0..1]"/>
  <xsd:element name="interval" type="xsd:WSDayIntervalType[0..1]"/>
  <xsd:element name="marketContext" type="emix:MarketContextType[0..1]"/>
  <xsd:element name="partyID" type="xsd:string"/>
  <xsd:element name="requestID" type="xsd:string"/>
  <xsd:element name="requestorPartyID" type="xsd:string"/>
  <xsd:element name="transactionID" type="xsd:string[0..1]"/>
</xsd:complexType>

<xsd:complexType name="EiReplyTransactionType">
  <xsd:element name="eiResponse" type="EiResponseType"/>
  <xsd:element name="eiTransaction" type="EiTransactionType[0..1]"/>
  <xsd:element name="responses" type="ArrayOfResponses[0..1]"/>
</xsd:complexType>
```
4.4 Post-Transaction Services

In a market of pure transactive energy, verification would be solely a function of meter readings. The seed standard for smart grid meter readings is the NAESB Energy Usage Information [NAESB EUI] specification.

In today’s markets, with most customers on Full Requirements tariffs, the situation is necessarily more complex. Full Requirements describes the situation where purchases are not
committed in advance. The seller is generally obligated to provide all that the buyer requires. Full requirements tariffs create much of the variance in today’s DR markets.

These sections will apply a measurement model consistent with the [NAESB EUI] as in the EIReport Services.

### 4.4.1 Energy Delivery Information

These service operations respond with Energy Usage Information or any other single item of interest to the caller. This is very simple, requesting one thing measured for one interval, and waiting to return a value until the information is available. For anything more complex the Report Services should be used.

#### Table 30-: Energy Delivery

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiDelivery</td>
<td>EiCreateDelivery</td>
<td>EiCreatedDelivery</td>
<td>Party</td>
<td>Party</td>
<td>Party-to-Party, specifying interval, what is to be measured, and the direction for the measurement</td>
</tr>
</tbody>
</table>

#### 2.4.1.1 Interaction Pattern for the EiDelivery Service

Figure 39-: Interaction Diagram for Delivery Service

#### 2.4.1.2 Information Model for the EiDelivery Service

The EiDelivery Type is a simplified EIReport.
Figure 40: UML of EiDelivery Type

2.4.1.3.4.1.3 Operation Payloads for the EiDelivery Service
4.5 Comparison of Transactive Payloads

Figure 41: UML Class Diagram of Delivery and Delivery Payload

Figure 42: UML Diagram comparing all Transactive Payloads
Enrollment is distinct from Registration in Energy Interoperation. Registration establishes an identity for an actor (a party or a device such as a generator or a meter on a premise). Enrollment establishes a relationship between two actors as a basis for further interactions. Energy Interoperation supports two classes of interactions; Transactive and VTN/VEN interactions.

In the case of enrollment in Transactive Interactions, the Enrollment Service identifies the two parties and the Enabling Agreement, Market, Tariff, Purchasing, Selling, etc. that the parties agree to use for their interactions.

In the case of enrollment in a VTN/VEN relationship the enrollment service identifies the two actors, generally a registered Resource and a Service Provider acting as a Designated Dispatch Entity (DDE). Registration of a Resource may sometimes be automatic with enrollment of the Resource.

The entities described in the following table can be enrolled. These are described in the [UML] diagrams as concrete classes that inherit from the Enrollee type. The strings are used to describe the entity; the standard approach to extensibility where a prefix of “x-” indicates an extension SHALL be used.

The types of entity used may depend on the implementation. All implementations SHALL support Resources.

<table>
<thead>
<tr>
<th>Entity</th>
<th>String</th>
<th>Description</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregator</td>
<td>aggregator</td>
<td>An entity that combines or aggregates generation or consumption</td>
<td></td>
</tr>
<tr>
<td>Consumer</td>
<td>customer</td>
<td>An entity that is generally a net consumer of electricity</td>
<td></td>
</tr>
<tr>
<td>Distribution</td>
<td>distribution</td>
<td>An entity that distributes electricity</td>
<td>E.g. a distribution utility</td>
</tr>
<tr>
<td>Enrolling Authority</td>
<td>enrollingAuthority</td>
<td>An entity that can perform enrolling services</td>
<td></td>
</tr>
<tr>
<td>Generator</td>
<td>generator</td>
<td>An entity that is generally a net producer of electricity</td>
<td></td>
</tr>
<tr>
<td>Load Serving Entity</td>
<td>ise</td>
<td>An entity which supports loads rather than generation</td>
<td></td>
</tr>
<tr>
<td>Market</td>
<td>market</td>
<td>A Market that enrolls in another Market Context</td>
<td></td>
</tr>
<tr>
<td>Meter Authority</td>
<td>meterAuthority</td>
<td>An entity that provides metering services</td>
<td></td>
</tr>
<tr>
<td>Resource</td>
<td>resource</td>
<td>An EMIX Resource with additional information</td>
<td>A Resource including performance envelope and additional information including Resource Name</td>
</tr>
<tr>
<td>Scheduling Entity</td>
<td>schedulingEntity</td>
<td>An entity that provides scheduling services</td>
<td></td>
</tr>
<tr>
<td>Service Provider</td>
<td>serviceProvider</td>
<td>An entity that provides services</td>
<td>A potential provider of services to the VTN in support of VTN business processes</td>
</tr>
<tr>
<td>Supplier</td>
<td>supplier</td>
<td>An entity that is generally a net supplier of electricity</td>
<td></td>
</tr>
<tr>
<td>System Operator</td>
<td>systemOperator</td>
<td>An entity that operates a grid</td>
<td></td>
</tr>
<tr>
<td>TDSP</td>
<td>tdsp</td>
<td>An entity which supports transmission and distribution of electricity</td>
<td></td>
</tr>
</tbody>
</table>
An entity which supports transmission of electricity

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiEnroll</td>
<td>EiCreateEnroll</td>
<td>EiCreatedEnroll</td>
<td>Party</td>
<td>Party</td>
<td>Create and send Enrollment</td>
</tr>
<tr>
<td>EiEnroll</td>
<td>EiRequestEnroll</td>
<td>EiReplyEnroll</td>
<td>Party</td>
<td>Party</td>
<td>Requests outstanding Enrollment information: request semantics with no time Interval</td>
</tr>
<tr>
<td>EiEnroll</td>
<td>EiCancelEnroll</td>
<td>EiCanceledEnroll</td>
<td>Party</td>
<td>Party</td>
<td>Cancel one or more Enrollments</td>
</tr>
</tbody>
</table>

### 5.1 Interaction Patterns for the EiEnroll Service

This is the [UML] interaction diagram for the EiEnroll Service.

![Interaction Diagram for the EiEnroll Service](image)

**Figure 43**: Interaction Diagram for the EiEnroll Service
5.2 Information Model for the EiEnroll Service

The EiEnroll service has an abstract class for the respective types. The abstract class also has the entity identifier, type (as a string), and name. The standard values for the type are listed in Table 31 Enrollee Descriptions. Other values MAY be used but MUST be prefixed by "x-" as described in Appendix Annex C.

Figure 44: UML Model for EiEnrollment Classes
5.3 Enrollee Types

The [UML] class diagram describes the Enrollee Types.

Figure 45: UML Class Diagram showing Enrollee Types
5.4 Operation Payloads for the EiEnroll Service

The [UML] class diagram describes the payloads for the EiEnroll service operations.

Figure 46: UML Class Diagram for Enrollment Payloads
### Event Services

The Event Service is used to call for performance under a transaction. The service parameters and event information distinguish different types of events. Event types include reliability events, emergency events, and more—and events MAY be defined for other actions under a transaction. For transactive services, two parties may enter into a call option. Invocation of the call option by the Promissee on the Promissor can be thought of as raising an event. But typically the Promissee may raise the event at its discretion as long as the call is within the terms of the call option transaction.

For example, an ISO that has awarded an ancillary services transaction to a Party may issue dispatch orders, which can also be viewed as Events. In this specification, what is sometimes called a *price event* would typically be communicated using the EiSendQuote operation (see 4.2).
Pre-Transaction Services”).

Table 33: Event Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response Operation</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiEvent</td>
<td>EiCreateEvent</td>
<td>EiCreatedEvent</td>
<td>VTN</td>
<td>VEN</td>
<td>Create and send a new Event</td>
</tr>
<tr>
<td>EiEvent</td>
<td>EiChangeEvent</td>
<td>EiChangedEvent</td>
<td>VTN</td>
<td>VEN</td>
<td>Modify an existing Event</td>
</tr>
<tr>
<td>EiEvent</td>
<td>EiRequestEvent</td>
<td>EiReplyEvent</td>
<td>Either</td>
<td>Either</td>
<td>Request outstanding Events, request semantics with optional time Interval</td>
</tr>
<tr>
<td>EiEvent</td>
<td>EiRequestPendingEvent</td>
<td>EiReplyPendingEvent</td>
<td>Either</td>
<td>Either</td>
<td>Similar to Request Events except that Reply returns Event IDs and Modification Numbers only.</td>
</tr>
<tr>
<td>EiEvent</td>
<td>EiCancelEvent</td>
<td>EiCanceledEvent</td>
<td>VTN</td>
<td>VEN</td>
<td>Cancel one or more Events</td>
</tr>
<tr>
<td>EiEvent</td>
<td>EiDistributeEvent</td>
<td></td>
<td>VTN</td>
<td>VEN</td>
<td>Broadcast of Event</td>
</tr>
</tbody>
</table>

### 6.1 Information Model for the EiEvent Service

The event is the core Demand Response information structure, and the most complex of the payloads. Understanding the information model of the Event is critical to understanding the operations of the Event Services. This section reviews the Event semantics as defined in Section 2.3 “Event-based Interactions”.

The sub-sections below provide a reprise of the Event structure (6.1.1) and a UML description of the event (6.1.2)

#### 6.1.1 Structure of the Event

The semantics of the Event are defined Section 2.3 “Event-based Interactions”.

![Event Model Diagram](image)
As the event is the core Demand Response information structure, we begin with Unified Modeling Language (UML) diagrams for the EiEvent class and for each of the operation payloads. Core semantics for the Event are defined in Section 2.3 "Event-based Interactions".
4.1.2 UML Model of an Event and its Signals

Figure 48: UML Class Diagram for EiEventType and Related Classes (w/o Signals detail)
An Event may include a number of Schedules, which are expressed as Streams. These schedules are the Signals, the Baselines, and they may return Baselines, Reports, and Delivery. The Ei Event Signal derives from the Streams element and conveys elements of the Type Signal Payload in its Schedule.

Figure 49- UML Class Diagram Showing Details of the Signal Payloads or EiEventSignals

6.2 Special Semantics of the Event Request Operations

The Events are the largest messages exchanged in Energy Interoperation. They exist in two forms, the EiEventRequest, and EiEventRequestPending. EiEventReply returns entire Events in response to a Request, following the general pattern of all Energy Interoperation Services. EiEventRequestPending returns the Event IDs and Modification Numbers only. EiEventRequestPending is useful for black-start and other situations in which the VEN and VTN need to assess the information shared with its partner.

The Modification Number returned in the Replies is for assessment only. The recipient MAY use it to determine that the sender is using out-of-date information, but any replacement or update SHALL convey the current Modification.

6.2.1 Event Ordering

The Event Requests include an option to restrict the number of Events returned in Reply to any Request. For consistency, this requires that a VTN or VEN be able to order Events. The rules for ordering Events are applied sequentially as follows:

1. Active events have priority over pending events
2. Within Active Events, priority is determined by Priority in the Event Descriptor.
3. Between active events with the same priority, the one with the earlier start time has the higher priority.
4. Between pending events the one with the earlier start time has the higher priority.
After processing rules 1-4, if Priority is still indeterminate within a set of Intervals, then
is indeterminate within that set. A Reply containing Events with indeterminate Order MUST
maintain that order in response to successive Requests while they remain indeterminate.

The definitions of Active and Pending are consistent with those described for the Event Filter
in Table 34.

### 6.2.2 Event Filter described

Both the Event Request operations MAY use of the Event Filter to restrict the Events exchanged
during Request and Reply.

#### Table 34: Event Filter described

<table>
<thead>
<tr>
<th>Event Filter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>An event qualifies if the Active Interval coincides with the Interval in the Request. An Event qualifies if any part of the Active Interval occurs within the specifying Interval; without accompanying Interval, “now” is treated as an infinitesimal Interval with a current starting date and time.</td>
</tr>
<tr>
<td>Pending</td>
<td>An event qualifies if the Active Interval starting date and time is in the future. If specified with an accompanying Interval, the Event qualifies if the Active Interval has not started (is not Active) at the Start of the Interval, and the Active Interval start is within the bounds of the specifying Interval.</td>
</tr>
<tr>
<td>All</td>
<td>An event qualifies if it would qualify as either Active or Pending.</td>
</tr>
<tr>
<td>Completed</td>
<td>An Event qualifies if the Active Interval is completed before the Request. If specified with an Interval in the Request, an Event qualifies if the end of the Active Interval occurs before the start of the Requested Interval. Conforming profiles MAY return a NULL set in response to a Request for Completed Intervals, as there is no requirement to store or be able to retrieve Completed Events.</td>
</tr>
<tr>
<td>Cancelled</td>
<td>An Event qualifies if it has been Cancelled. If specified with an accompanying Interval, and Event qualifies if the Event would have qualified as Active during the Interval. Conforming profiles MAY return a NULL set in response to a request for Completed Intervals as there is no requirement to store or be able to retrieve Cancelled Events.</td>
</tr>
</tbody>
</table>

#### 6.2.3 Using EiRequestEvent EiRequestEventPending together

EiRequestEvent and EiRequestEventPending are essentially the same. Each enables a VEN or
VTN to query its partner about what Events it knows. The difference is in the Replies. EiReplyEvent returns a collection of Qualified Event IDs, i.e., an Event ID and the Modification Number.

**Figure 50: Qualified Event ID**

With a list of Qualified Event IDs either one can reconstruct what the other knows. Events that
are missing can be requested or sent. A VEN can infer cancellation when its VTN removes an
Event ID. Using the Modification Number, a VTN can know to re-send the latest version, or a
VEN can know to request an update.

While the Event Requests follow the pattern common to all EI Requests, because of the extra
options, they are summarized in table [reference] below. All query elements are optional.
Table 35: Event Requests summarized

<table>
<thead>
<tr>
<th>Request Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEN ID</td>
<td>Names the VEN that is Requesting or currently knows of these Events</td>
</tr>
<tr>
<td>Event ID</td>
<td>A list of Event IDs to be returned. If present, all other filters are ignored.</td>
</tr>
<tr>
<td>Market Context</td>
<td>Request is to return Events that are in a Market Context. For example, in a given Program, a VEN could request all Electric Vehicle (EV) related Events.</td>
</tr>
<tr>
<td>Filter</td>
<td>As described above (Table 34). Can be combined with Interval</td>
</tr>
<tr>
<td>Interval</td>
<td>Requests Events “within” an Interval. Interval may contain only a Start Date to request all Events from that date forward, or may include only an End Date to include events before that Date. If no Interval is present, this is interpreted as if the Filter were “all”.</td>
</tr>
<tr>
<td>Reply Limit</td>
<td>Return only the first N matching events, where N is the Reply Limit. “First” is defined according to the Order as described above.</td>
</tr>
</tbody>
</table>

A common pattern for either a VEN or a VTN is to request Event IDs with the EiRequestPending, and to then request information about events that it are missing or that need updates using EiRequestEvent. A VTN after a similar query might use EiCreateEvent to pass the missing or updated Events to the VEN.
6.3 Interaction Patterns for the EiEvent Service

This is the [UML] interaction diagram for the EiEvent Service.

Figure 51: UML Interaction Diagram for the EiEvent Service Operations
Figure 52: UML for example PULL pattern for EiEvent

Figure 53: Interaction Diagram for Pending Event operation
### 6.4 Operation Payloads for the EiEvent Service

The [UML] class diagram describes the payloads for the EiEvent service operations.

![UML Class Diagram for EiEvent Service Operation Payloads](image)

Figure 54: UML Class Diagram for EiEvent Service Operation Payloads
Energy Interoperation Reports convey information from remote sensing or about remote state back to the requester. The Historian operations support the collection of data for Reports. Reports can be associated with an Event or can be requested through the Report Services described in this section.

The general pattern of the Report service is to request that a Historian gather data, and for the Report Service to return the Report when it is Ready. A Historian may generate only a final Report, or it may report-back periodically. The report requester MAY ask the Historian for the report-to-date, or for a time-constrained portion of the Report at any time while it is running.

One interaction pattern for the Report service is what one may call "Set and Forget". Under this pattern, the Requester asks that information be logged, but specifies no Report delivery. Under this pattern, the Requester can, at any time, request delivery of a Report for a specified interval.

Projections are a special class of Reports, i.e., Reports about the future. Projections follow the general form of Reports and include additional metadata about the reliability of the future information in each window.

The semantics of Reports are described in sections 2.4 "Monitoring, Reporting" and 2.5 "Reports, Snaps, and Projections".

The range of Payloads that can be delivered by means of a Report can be extended by deriving new types from the Payload Base Type, and defining a new Report Type not in Enumerated Report Types, and requesting such a Report.

### 7.1 Overview of Report Services

Event-based reports are requested as part of the EiEvent service. Ei Report operations request Reports independently of any Event. Whether created as part of an Event or independently, all Reports support the same post-creation operations.

EiReport operations are independent of EiEvent operations in that they can be requested at any time independent of the status or history of EiEvents.

<table>
<thead>
<tr>
<th>Table 36: Report Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
<tr>
<td>EiReport</td>
</tr>
</tbody>
</table>
7.2 EiHistorian Service

7.2.1 Interaction Pattern for the EiHistorian Service

Figure 55: Interaction Pattern for Historian Operations (Report Service)
### 7.2.2 Operations Payloads for the EiHistorian Service

#### UML Diagram of Historian Payloads

![UML Diagram of Historian Payloads](image)

---

**Figure 56**: UML Diagram of Historian Payloads
7.3 EiReport Service

7.3.1 Information Model for the EiReport Service

An EiReport is prepared by a Party upon request and supplied to the requesting party. It may also be defined in the expectations of the Market Context.

Figure 57: UML Class Diagram for the EiReport Class
7.3.2 Interaction Pattern for the EiReport Service

This is the [UML] interaction diagram for the EiReport Service.

Figure 58: UML Interaction Diagram for the EiReport Service (Report Service)
7.3.3 Operation Payloads for the EiReport Service

Figure 59: UML Diagram of Report Payloads

- `<XSDcomplexType>`
  - `<XSDelement>`
    - eReportRequest: EiReportRequestType [1..*]
    - partyID: actorID
    - requestID: refID
    - requestorPartyID: actorID
    - vRefID: actorID [0..1]

- `<XSDcomplexType>`
  - `<XSDelement>`
    - eReport: EiReportType [0..*]
    - eResponse: EiResponseType
    - responses: ArrayOfResponses [0..1]

- `<XSDcomplexType>`
  - `<XSDelement>`
    - EiCreateReportType

- `<XSDcomplexType>`
  - `<XSDelement>`
    - EiUpdateReportType

- `<XSDcomplexType>`
  - `<XSDelement>`
    - EiCancelReportType

- `<XSDcomplexType>`
  - `<XSDelement>`
    - EiCanceledReportType
7.4 _EiProjectionService_

7.4.1 Interaction Pattern for EiProjection Service

Figure 60: Interaction Pattern for Projection Operations (Report Service)
7.4.2 Operation Payloads for the EiProjection Service

**Figure 61**: UML Diagram of Projection Payloads
7.5 Summary of Report Payloads

The [UML] class diagram below recaps the payloads for all operations of the EiReportService.

![UML Class Diagram for all EiReportService Operation Payloads]

Figure 62.: UML Class Diagram for all EiReportService Operation Payloads
Users of [OpenADR] found that they needed to be able to constrain the application of remote DR services. For the DR Operator, advanced knowledge of these constraints improved the ability to predict results. The services in this section are based on the services used to tailor expectations in [OpenADR].

Availability and Opt are similar in that they communicate when a Party is willing to receive an Event. Availability is a long-term schedule for when a Party will consider a response. Availability could be set in the Market Context or at program enrollment. Opt (as in opt in or opt out) encompasses short-term additions to or replacement of the schedule in Availability.

The combination of Availability and Opt states together define the times during which a committed response from the VEN is possible or likely.

8.1 Relationship of Availability and Opt Information
Availability and Opt apply to interactions where an action is requested (e.g. curtailment and DER actions), and only indirectly to (e.g.) price distribution interactions.

Availability is a long-term description and may be complex. Opt is a short-term description that replaces or is combined into the long-term availability description.

Availability and Opt-In and Opt-Out, as well as Market Rules, use the VavailabilityType defined in [WS-Calendar] which in turn is an XML serialization of [Vavailability]. The semantics are defined in [Vavailability].

The behavior of the Availability schedule is defined as follows. We call the parameter passed for Opt-In and Opt-Out the Vavailability.

- The EiAvailability class describes when the VEN expects/plans to be available to respond to a request for performance, generally an EiEvent.
- Exactly one Vavailability is included in the EiAvailability and the EiOpt objects.
- An EiOpt that is used in a message MUST have a bounded interval (the Opt Interval) in the Vavailability.
- An Opt-In while in effect adds the available times of the Vavailability to the available times in the bounded interval for the VEN with respect to a MarketContext, effectively performing a logical OR operation on the available times but only within the opt Interval.
- An Opt-Out while in effect replaces the entire portion of the EiAvailability within the opt Interval.
- Exactly zero or one Opt functions MAY be in effect at any time.

In short, Opt-In adds the Vavailability available times to the overall VEN availability; Opt-Out replaces the entirety of its opt intervals with the contents of the Opt-Out Vavailability.

8.2 EiAvail Service
The Availability is set by the VEN and indicates when an event may or may not be accepted and executed by the VEN with respect to a Market Context. Knowing the Availability and Opt information for its VENs improves the ability of the VTN to estimate response to an event or request.

---

3 By defining an end time for the Vavailability
4 Called Constraints in [OpenADR1]
When Availability is set, opt-in or opt-out does not affect the Availability except for the specific interval(s) described by the Opt—opting out is temporary unavailability, which may have transaction and business consequences if an event is created during the opt-out period.

The modeling for Availability includes behavior indications for the situation where an EiEvent overlaps a constrained time interval.

EiAvailability describes only the available times, using the patterns defined in [WS-Calendar] and [Vavailability].

### Table 37: Avail Service

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiAvail</td>
<td>EiCreateAvail</td>
<td>EiCreatedAvail</td>
<td>VEN</td>
<td>VTN</td>
<td>Create an Avail for this VEN; return the AvailID</td>
</tr>
<tr>
<td>EiAvail</td>
<td>EiRequestAvail</td>
<td>EiReplyAvail</td>
<td>VEN</td>
<td>VTN</td>
<td>Request Avail information for this VEN; request semantics with no time Interval</td>
</tr>
<tr>
<td>EiAvail</td>
<td>EiCancelAvail</td>
<td>EiCanceledAvail</td>
<td>VEN</td>
<td>VTN</td>
<td>Cancel the Avail referenced by the AvailID</td>
</tr>
</tbody>
</table>

The element EiAvailBehavior defines how an issued EiEvent that conflicts with the current EiAvail is performed:

- **ACCEPT** – accept the issued EiEvent regardless of conflicts with the EiAvail
- **REJECT** – reject any EiEvent whose schedule conflicts with the EiAvail
- **RESTRICT** – modify the EiEvent parameters so that they fall within the bounds of the EiAvail

#### 3.2.1 Interaction Patterns for the EiAvail Service

This is the [UML] interaction diagram for the EiAvail Service.

![Interaction Diagram for EiAvail Service](image)

**Figure 63:** Interaction Pattern for the EiAvailability Service.
### 8.2.2 Information Model for the EiAvail Service

![UML Class Diagram for the EiAvail Type](image)

**Figure 64:** UML Class Diagram for the EiAvail Type
### 3.2.3 Operation Payloads for the EiAvail Service

The [UML] class diagram describes the payloads for the EiAvail service operations.
8.3 EiOpt Service

The Opt service creates and communicates Opt-In and Opt-Out schedules from the VEN to the VTN. Schedules are combined with EiAvailability and the Market Context requirements to give a complete picture of the willingness of the VEN to respond to EiEvents received by the VEN.

- Exactly one Vavailability MUST be provided in EiCreateOptIn and EiCreateOptOut.
Opt schedules SHALL override any Availability in place while there is an Opt in effect.

Section 8.1

Applying EiCreateOptIn or EiCreateOptOut if an Opt is currently in effect replaces the current Opt in effect with that in the Opt Vavailability, which effectively cancels the current Opt state and Creates a new one.

### Table 38: Opt Service

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiOpt</td>
<td>EiCreateOpt</td>
<td>EiCreatedOpt</td>
<td>VEN</td>
<td>VTN</td>
<td>Create and send an Opt, receiving an Opt ID</td>
</tr>
<tr>
<td>EiOpt</td>
<td>EiRequestOpt</td>
<td>EiReplyOpt</td>
<td>VEN</td>
<td>VTN</td>
<td>Request the Opts from the VTN that are currently in effect, at most one per Market Context.</td>
</tr>
<tr>
<td>EiOpt</td>
<td>EiCancelOpt</td>
<td>EiCanceledOpt</td>
<td>VEN</td>
<td>VTN</td>
<td>Cancel the identified Opt</td>
</tr>
</tbody>
</table>

### 8.3.1 Interaction Patterns for the EiOpt Service

This is the [UML] interaction diagram for the EiOpt Service.

![Interaction Diagram for the EiOpt Service](image)

### 8.3.2 Information Model for the EiOpt Service

Opting in or out is a temporary situation indicating that the VEN will or will not respond to an event or in a specific time period, without changing the potentially complex Availability. The EiOpt schedule is a [WS-Calendar] VavailabilityType.
3.3.3 Operation Payloads for the EIOpt Service

The [UML] class diagram describes the payloads for the EIOpt service operations.
Figure 68: UML Class Diagram for EiOpt Service Operation Payloads
Market Information

Each Event and Service in Energy Interoperation takes place within a Market Context. This Context defines the behaviors that each Party can expect from the other.

9.1 The Market Context

Market Contexts are used to express market information that rarely changes, and thereafter not need to communicate it with each message.

In any market context, there are standing terms and expectations about product offerings. If these standing terms and expectations are not known, many exchanges may need to occur before finding products that meet those expectations. If these expectations are only known through local knowledge, then national and international products need to be re-configured for each local market that they enter. If all market information were to be transmitted in every information exchange, messages based on EMIX would be overly repetitious.

As described in Section 2.2 “Market Context”, The EI Market Contexts is a super-set of the [EMIX] Standard Terms, and they can be referenced using the EMIX Market Context as an identifier. The EMIX Market Context is expressed as an URI.

9.2 Market Context Service

The Market Context Service enables a Party to request the details of a Market Context. These MAY be mandatory in many of today’s interactions. Parties MAY be able to request and compare Market Contexts to select which markets to participate in. Such Interactions are out of scope for this specification.

![Sequence diagram for Market Context service](image)

The Market Context service can retrieve the full information in an EIMarketContext given the identifier, an EMIX Market Context. There is one operation and a responding operation.

**Table 39: Market Context Service**

<table>
<thead>
<tr>
<th>Service</th>
<th>Operation</th>
<th>Response</th>
<th>Service Consumer</th>
<th>Service Provider</th>
<th>Notes</th>
</tr>
</thead>
</table>
9.3 Information Model for the EiMarketContext Service

Figure 70: UML Class Diagram for Market Context

9.4 Operation Payloads for the EiMarket Context Service

Figure 71: UML of Market Context Service payloads
2.10 Security and Composition [Non-Normative]

This section describes the enterprise software approach to security and composition as applied to this Energy Interoperation specification.

Service orientation has driven a great simplification of interoperation, wherein software is no longer based on Application Programming Interfaces (APIs) but is based on exchange of information in a defined pattern of services and service operations [SOA-RM].

The approach for enterprise software has evolved to defining key services and information to be exchanged, without definitively specifying how to communicate with services and how to exchange information—there are many requirements for distributed applications in many environments that cannot be taken into account in a service and information standard. To make such choices is the realm of other standards for specific areas of practice, and even there due care must be taken to avoid creating a monoculture of security.\(^5\)

10.1 Security and Reliability Example

Different interactions require different choices for security, privacy, and reliability. Consider the following set of specifics. (This figure is here repeated and re-labeled.)

---

5 See e.g. the STUXNET worm effects on a monoculture of software SCADA systems, 2010. See http://en.wikipedia.org/wiki/Stuxnet


---

Figure 72a: Web of Example DR Interactions

We specifically model a Reliability DR Event initiated by the Independent System Operator\(^6\) A, who sends a reliability event to its first-level aggregators B through E. Aggregator B, in turn invokes the same service on its customers (say real estate landlords) F, G, and H.

Those customers might be industrial parks with multiple facilities, real estate developments with multiple tenants, or a company headquarters with facilities in many different geographical areas, which would invoke the same operation on their VENs.
For our example, say that G is a big-box store regional headquarters and I, J, and L are the stores in the affected area.

Each interaction will have its own security and reliability composed as needed—the requirements vary for specific interactions. For example:

- For service operations between A to B, typical implementations include secure private relay networks with guaranteed high reliability and known latency. In addition, rather than relying on the highly reliable network, in this case A requires an acknowledgment message from B back to A proving that the message was received.

- From the perspective of the ISO, the communication security and reliability between F and its customers F, G, and H may be purely the responsibility of B, who in order to carry out B’s transaction commitments to A will arrange its business and interactions to meet B’s business needs.

- G receives the signal from aggregator B. In the transaction between G and B, there are service, response, and likely security and other requirements. To meet its transactional requirements, the service operations between B and G will be implemented to satisfy the business needs of both B and G. For our example, they will use the public Internet with VPN technology and explicit acknowledgement, with a backup of pagers and phone calls in the unlikely event that the primary communication fails. And each message gets an explicit application level acknowledgement. Security between B and G depends on the respective security models and infrastructure supported by B and G—no one size will fit all. So that security will be used for that interaction.

- The big box store chain has its own corporate security architecture and implementation, as well as reliability that meets its business needs—again, no one size will fit all, and there is tremendous variation; there is no monoculture of corporate security infrastructures.

- Store L has security, reliability, and other system design and deployment needs and implementations within the store. These may or may not be the same as the WAN connection from regional headquarters G, and in fact are typically not the same (although some security aspects such as federated identity management and key distribution might be the same).

- Store L also has a relationship with aggregator E, which for this example is Store L’s local utility; the Public Utility Commission for the state in which L is located has mandated (in this example) that all commercial customers will use Energy Interoperation to receive certain mandated signals and price communications from the local utility. The PUC, the utility, and the owner of the store L have determined the security and reliability constraints. Once again, one size cannot fit all—and if there were one “normal” way to accommodate security and reliability, there will be a different “normal” way in different jurisdictions.

So for a simple Demand Response event distribution, we have potentially four different security profiles.

The following table has sample functional names for selected nodes.

<table>
<thead>
<tr>
<th>Label</th>
<th>Structure Role</th>
<th>Possible Actor Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>VTN</td>
<td>System Operator</td>
</tr>
<tr>
<td>B</td>
<td>VEN (wrt A), VTN (wrt F, G, H)</td>
<td>Aggregator</td>
</tr>
<tr>
<td>G</td>
<td>VEN (wrt B), VTN (wrt I, J, L)</td>
<td>Regional Office</td>
</tr>
<tr>
<td>L</td>
<td>VEN (wrt G and wrt E)</td>
<td>Store</td>
</tr>
<tr>
<td>E</td>
<td>VEN (wrt A, VTN wrt L)</td>
<td>Local Utility</td>
</tr>
</tbody>
</table>

(Note: wrt means “with respect to”)
10.2 Composition

In state-of-the-art software architecture, we have moved away from monolithic implementations and standards to ones that are composed of smaller parts. This allows the substitution of a functionally similar technology where needed, innovation in place, and innovation across possible solutions.

In the rich ecosystem of services and applications in use today, we compose or (loosely) assemble applications rather than craft them as one large thing. See for example OASIS Service Component Architecture [OASIS SCA], which addresses the assembly, substitution, and independent evolution of components.

A typical web browser or email system uses many standards from many sources, and has evolved rapidly to accommodate new requirements by being structured to allow substitution. The set of standards (information, service, or messaging) is said to be composed to perform the task of delivery of email. Rather than creating a single application that does everything, perhaps in its own specific way, we can use components of code, of standards, and of protocols to achieve our goal. This is much more efficient to produce and evolve than large integrated applications such as older customized email systems.

In a similar manner, we say we compose the required security into the applications—say an aspect of OASIS [WS-Security] and OASIS Security Access Markup Language [SAML]—and further compose the required reliability, say by using OASIS [WS-ReliableMessaging] or perhaps the reliable messaging supported in an Enterprise Service Bus that we have deployed.

A service specification, with specific information to be exchanged, can take advantage of and be used in many different business environments without locking some in and locking some out, a great benefit to flexibility, adoption, and re-use.

10.3 Energy Interoperation and Security

In this section we describe some specific technologies and standards in our palette for building a secure and reliable implementation of Energy Interoperation. Since Energy Interoperation defines only the core information exchanges and services, and other technologies are composed in, there is no optionality related to security or reliability required or present in Energy Interoperation.

The information model in Energy Interoperation 1.0 is just that—an information model without security requirements. Each implementation must determine the security needs (outside the scope of this standard) broadly defined, including privacy (see e.g. OASIS Privacy Management Reference Model [PMRM]), identity (see e.g. OASIS Identity in the Cloud, OASIS Key Management Interoperability, OASIS Enterprise Key Management Infrastructure, OASIS Provisioning Services, OASIS Web Services Federation TC, OASIS Web Services Secure Exchange and more)

Energy Interoperation defines services together with service operations, as is now best practice in enterprise software. The message payloads are defined as information models, and include such artifacts as Energy Market Information Exchange [EMIX] price and product definition, tenders, and transactions, the EiEvent artifacts defined in this specification, and all information required to be exchanged for price distribution, program event distribution, demand response, and distributed energy resources.

This allows the composition and use of required interoperation standards without restriction, drawing from a palette of available standards, best practices, and technologies. The requirements to be addressed for a deployment are system issues and out of scope for this specification.

As in other software areas, if a particular approach is commonly used, then a separate standard (or standardized profile) may be created. In this way, WS-SecureConversation composes WS-Reliability and WS-Security.

So Energy Interoperation defines the exchanged information, the services and operations, and as a matter of scope and broad use does not address any specific application as the security
privacy, performance, and reliability needs cannot be encompassed in one specification. Many
of the TCs named above have produced OASIS Standards,

### 3.11 Profiles [Normative]

These sections define the three normative profiles that are part of Energy Interoperation 1.0.

A profile includes a selection of interfaces, services, and options for a particular purpose.

#### 3.11.1 OpenADR [Normative]

The OpenADR Profile defines the services required to implement functionality similar to that in [OpenADR]. The inclusion of the Energy Interoperation structure of VTNs and VENs, as well as use of the Energy Market Information Exchange [EMIX] cross-cutting price and product definition standard and WS-Calendar [WS-Calendar] based on the IETF [iCalenar] RFC updates and gives a broader range of applicability in what has been described as the OpenADR 2 Profile.

We present in simplified tabular form the Energy Interoperation services required as part of the OpenADR Profile. When a service is included, all of the listed operations are included, so we list only the service name and the section of this document.

Table 41: Services used in OpenADR Profile

<table>
<thead>
<tr>
<th>Service</th>
<th>Section</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiRegisterParty</td>
<td>4.1</td>
<td>Register to identify and receive information</td>
</tr>
<tr>
<td>EiQuote</td>
<td>4.2</td>
<td>EIDistributeQuote for distributing dynamic prices (push), other operations for pull including block and tier tariff communication</td>
</tr>
<tr>
<td>EiEvent</td>
<td>6</td>
<td>The core event functions and information models</td>
</tr>
<tr>
<td>EiReport</td>
<td>7</td>
<td>The ability to set periodic or one-time information on the state of a Resource</td>
</tr>
<tr>
<td>EiAvail</td>
<td>8.2</td>
<td>Constraints on the possible time a Resource is available or not</td>
</tr>
<tr>
<td>EiOpt</td>
<td>8.3</td>
<td>Overrides the EiAvail; addresses short-term changes in availability</td>
</tr>
<tr>
<td>EiEnroll</td>
<td>5</td>
<td>Used to enroll a Resource for participation in Events.</td>
</tr>
<tr>
<td>EiMarketContext</td>
<td>9.2</td>
<td>Used to discover program rules, standard reports, etc.</td>
</tr>
</tbody>
</table>

#### 3.11.2 TeMIX [Normative]

The Transactive EMIX (TeMIX) Profile defines the services required to implement functionality for energy market interactions.

We present in simplified tabular form the Energy Interoperation services required as part of the TeMIX Profile. When a service is included, all of the listed operations are required, so we list only the service name and the section of this document.

Table 42: Services used in TeMIX Profile

<table>
<thead>
<tr>
<th>Service</th>
<th>Section</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiRegisterParty</td>
<td>4.1</td>
<td>Register to identify and receive information</td>
</tr>
<tr>
<td>EiQuote</td>
<td>4.2</td>
<td>EIDistributeQuote for distributing dynamic prices (push), other components for pull</td>
</tr>
<tr>
<td>EiTender</td>
<td>4.2</td>
<td>The basic offer of agreement is called a tender</td>
</tr>
<tr>
<td>EiTransaction</td>
<td>4.3</td>
<td>The core services to reach agreement</td>
</tr>
<tr>
<td>EiEnroll</td>
<td>5</td>
<td>Used to enroll a Resource for participation in Events.</td>
</tr>
<tr>
<td>EiMarketContext</td>
<td>9.2</td>
<td>Used to discover program rules, standard reports, etc.</td>
</tr>
<tr>
<td>EiDelivery</td>
<td>4.4.1</td>
<td>Post-Transaction delivery information</td>
</tr>
</tbody>
</table>

#### 3.11.3 Price Distribution [Normative]

Many current initiatives envision Price Distribution as a separate Profile requiring neither transactive energy nor event-based interactions. The Price Distribution profile defines the minimal set of services required to interact with a pure Price Distribution context.
We present in simplified tabular form the Energy Interoperation services required as part of the Price Distribution Profile. When a service is included, all of the listed operations are required, so we list only the service name and the section of this document.

<table>
<thead>
<tr>
<th>Service</th>
<th>Section</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>EiRegisterParty</td>
<td>4.1</td>
<td>Register to interact with other Parties</td>
</tr>
<tr>
<td>EiQuote</td>
<td>4.2</td>
<td>EiDistributeQuote for distributing dynamic prices (push), other components for pull</td>
</tr>
<tr>
<td>EiEnroll</td>
<td>5</td>
<td>Used to enroll in a Market to receive Price Distribution.</td>
</tr>
<tr>
<td>EiMarketContext</td>
<td>9.2</td>
<td>Used to discover program rules, standard terms, etc.</td>
</tr>
</tbody>
</table>
Conformance and Processing Rules for Energy Interoperation

12.1 Conformance for Energy Interoperation

We define four conformance points for Energy Interoperation 1.0, modified by the networking technology used

- Full Conformance
- Conformance

And further define

- Conformance to a Named Profile
- Conformance with Alternate Interoperation

In this section Named Profile is one of the profiles defined in Section 11.

12.2 General Conformance Requirements

The version of Energy Interoperation to which conformance is claimed MUST be specified in the implementation’s conformance statement.

Any extension(s) used by the implementation, whether of information structures, services, service operations, or payloads MUST be described in the Implementation’s conformance statement including the service operations, payloads, and information artifacts.

The phrase “support all XML artifacts” includes the support of XML artifacts as extended; similarly, message headers (SOAP Headers for Web services) MAY be extended as needed to compose other technologies including but not limited to reliability and security. The payloads defined in this specification are for required information exchanges, and a Conforming implementation MAY extend the data types, payloads, or message headers appropriate to their transport/networking as necessary. It is required that those extensions, restrictions, and so forth be documented in the conformance statement.

12.2.1 Full Conformance to Energy Interoperation

An implementation claiming Full Conformance to Energy Interoperation 1.0 MUST do all of the following as defined in this Work Product including specification, schemas, and WSDL files:

- Implement all services and service operations (“Services and Operations”)
- Support all XML artifacts as defined in the schemas (“XML”)
- Interoperate using Web services and the [WSDL] files (“Web Services Interoperation”)
- Conform to all of Sections 13.2 through the end of Section 13 (“Additional Conformance”)
- Describe how any relevant XML artifacts are derived from the Work Product

It is RECOMMENDED that interoperation be achieved using the WSI Basic Profile [WSI-Basic]

12.2.2 Conformance to Energy Interoperation

An implementation claiming Conformance to Energy Interoperation 1.0 MUST do all of the following as defined in this Work Product including specification, schemas, and WSDL files:

- Interoperate using Web services and the [WSDL] files (“Web Services Interoperation”)
- Conform to all of Sections 13.2 through the end of Section 13 (“Additional Conformance”)
- Describe how any relevant XML artifacts are derived from the Work Product
In addition, if the application claiming conformance does not support one or more Services or Operations as defined in this specification, then the conformance statement for the implementation must:

1. List all Services and Operations that are supported in the implementation.
2. List all Services and Operations that are not supported in the implementation.
3. For each Operation that is not supported, define the error response that will be returned upon invocation.

For those operations that are supported by an implementation, but whose use or semantics are restricted, a conforming implementation SHALL

1. List the subset of XML artifacts as defined by the schemas used in the implementation.
2. List the subset of XML artifacts as defined by the schemas that are not used in the specification.
3. State any restrictions, i.e., in cardinality or optionality, that is applied to artifacts defined herein.

### 12.2.3 Full Conformance with Alternate Interoperation to Energy Interoperation

An implementation claiming Full Conformance with Alternate Interoperation to Energy Interoperation 1.0 MUST be able to claim Full Conformance to Energy Interoperation, except that networking technologies other than Web services MAY be used by the implementation. A description of networking technologies used MUST be included in the implementation’s conformance statement.

An implementation MAY claim Full Conformance as well as Full Conformance with Alternate Interoperation. The Conformance statement MUST describe the extensions or departures from Full Conformance.

### 12.2.4 Conformance with Alternate Interoperation to Energy Interoperation

An implementation claiming Conformance with Alternate Interoperation to Energy Interoperation 1.0 MUST be able to claim Conformance to Energy Interoperation, except that networking technologies other than Web services MAY be used by the implementation. A description of networking technologies used MUST be included in the implementation’s conformance statement.

An implementation MAY claim Conformance as well as Conformance with Alternate Interoperation. The Conformance statement MUST describe the extensions or departures from Full Conformance.

### 12.2.5 Conformance to Named Profiles of Energy Interoperation

In this section Named Profile refers to one of the profiles defined in Section 11.
Profiles [Normative]*.

4.1.6.12.2.5.1 Full Conformance to a Named Profile of Energy Interoperation
An implementation claiming Full Conformance to a Named Profile of Energy Interoperation MUST be able to claim Full Conformance to Energy Interoperation excepting only the following:

- Services and Operations in sections not included in the named Profile as defined in Section 14 [wd35 – should be link]

It is RECOMMENDED that Web services interoperation be achieved using the WSI Basic Profile [WSI-Basic]

4.1.6.212.2.5.2 Conformance to a Named Profile of Energy Interoperation
An implementation claiming Conformance to a Named Profile of Energy Interoperation MUST be able to claim Conformance to Energy Interoperation excepting only the following:

- Services and Operations in sections not included in the named Profile

It is RECOMMENDED that Web services interoperation be achieved using the WSI Basic Profile [WSI-Basic]

4.1.6.312.2.5.3 Full Conformance or Conformance with Alternate Interoperation to a Named Profile
An implementation claiming Conformance with Alternate Interoperation or Full Conformance with Alternate Interoperation to a Named Profile of Energy Interoperation MUST be able to claim the respective Full Conformance or Conformance with Alternate Interoperation to Energy Interoperation excepting only the following:

- Services and Operations in sections not included in the Named Profile

In addition, interoperation payloads MUST be used as defined or extended; in the event that payloads are extended a description of the extension(s) SHALL be included in the Implementation’s conformance statement.

12.3 Conformance with the Semantic Models of EMIX and WS-Calendar
This section specifies conformance with the semantic models of [EMIX] and [WS-Calendar]. Energy Interoperation is strongly dependent on each of these information models.

[WS-Calendar] is a general specification and makes no assumptions about how its information model is used. [WS-Calendar] has specific rules which define Inheritance as a means to reduce the conveyance of repetitive information. As this specification constrains schedule communications to specific business interactions, these inheritance rules are extended to embrace rules of interaction and rules of process that further reduce the information that must be expressed in each interval.

Implementations of Energy Interoperation SHALL conform to the rules of [WS-Calendar] and [EMIX]. These rules include the following conformance types:

- Conformance to the inheritance rules in [WS-Calendar], including the direction of inheritance
- Specific attributes for each type that MUST or MUST NOT be inherited.
- Conformance rules that Referencing Specifications MUST follow
- Description of Covarying attributes with respect to the Reference Specification
- Semantic Conformance for the information within the Artifacts exchanged.
Conformance to the *inheritance rules* in [EMIX], including inheritance of Product Definitions and Standard Terms.

Energy Interoperation implementations also use the EMIX Products and Resources also extend the Inheritance patterns of [WS-Calendar] as specified in the EMIX information model. We address each of these in the following sections.

### Recapitulation of Requirements from WS-Calendar and EMIX

[WS-Calendar] uses the term Sequence to refer to one or more Intervals with Temporal Relations defined between them that may inherit from zero or more Gluons. [EMIX] introduced the term Schedule to refer to Product Descriptions applied to a Sequence. Streams recapitulate these rules with specific addenda as they include both Gluon and Sequence.

#### Specific Attribute Inheritance within Schedules

The rules that define inheritance, including direction in [WS-Calendar], are recapitulated.

**I1: Proximity Rule** Within a given lineage, inheritance is evaluated through each Parent to the Child before what the Child bequeaths is evaluated.

**I2: Direction Rule** Intervals MAY inherit attributes from the nearest Gluon subject to the Proximity Rule and Override Rule, provided those attributes are defined as Inheritable.

**I3: Override Rule** If and only if there is no value for a given attribute of a Gluon or Interval, that Gluon or Interval SHALL inherit the value for that attribute from its nearest Ancestor in conformance to the Proximity Rule.

**I4: Comparison Rule** Two Sequences are equivalent if a comparison of the respective Intervals succeeds as if each Sequence were fully Bound and redundant Gluons are removed.

**I5: Designated Interval Inheritance** [To facilitate composition of Sequences] the Designated Interval in the ultimate Ancestor of a Gluon is the Designated Interval of the composed Sequence. Special conformance rules for Designated Intervals apply only to the Interval linked from the Designator Gluon.

**I6: Start Time Inheritance** When a start time is specified through inheritance, that start time is inherited only by the Designated Interval; the start times of all other Intervals are computed through the durations and temporal relationships within the Sequence. The Designated Interval is the Interval whose parent is at the end of the lineage. In Events, the Active Interval is the Designated Interval.

#### Time Zone Specification

The time zone MUST be explicitly known in any conforming Energy Interoperation artifact. This may be accomplished in two ways:

- The time, date, or date and time MUST be specified using [ISO8601] utc-time (also called zulu time)

- The [WS-Calendar] Time Zone Identifier, TZID, MUST be in the Lineage of the artifact, as extended by the Market Context. Generally, the Market Context acts as a Gluon bequeathing the TZID. See Section 12.4 below.

If neither expression is included, the Artifact does not conform to this specification and its attempted use in information exchanges MUST result in an error condition.

#### Specific Rules for Optimizing Inheritance

If the Designated Interval in a Series has a Price only, all Intervals in the Sequence have a Price only and there is no Price in the Product.

- If the Designated Interval in a Series has a Quantity only, all Intervals in the Sequence have a Quantity only and there is no quantity in the Product.
• If the Designated Interval in a Series has a Price & Quantity, all Intervals in the Sequence MUST have a Price and Quantity and there is neither Price not Quantity in the Product.

### 12.4 TeMIX Conformance

The TeMIX Profile MUST apply the conformance rules for TeMIX described in [EMIX].

### 12.5 Inheritance within Events

For purposes of processing, inheritance, and conformance, Signal Information is treated as an [EMIX] Product Description, applied to a Sequence, and the Active Period is considered as a [WS-Calendar] Schedule. The Streams in Signals and Event-linked Reports inherit from the Active Interval as if it were a Gluon.

Signals within an Event arrive in a setting established by a Market Context. Within an event, there may be multiple Signal types. For purposes of inheritance, An Event may include multiple Stream-derived information elements each with an associated Sequence. For purposes of processing, the body of the Stream is treated as a [WS-Calendar] Gluon, and the Signal Information in each interval in the Sequence inherits from that Gluon.

Each Specifies a Market Context. If that Market Context is associated with Standard Terms, then those Terms enter the Lineage of the Schedule and are inherited by each Interval. Standard Terms associated with a Market Context enter the Lineage of the Schedule as if the Market Context were a Gluon. Product Description, TZID, Level Definition, Terms, et al. can be inherited in this way.

### 12.5.1 Sequence Optimization within Events

As described in 1.3.2 “Conformance of Streams to WS-Calendar”, Signals, Reports, and Baselines MUST conform to WS-Calendar.

### 12.6 Version Conformance

Implementations that use the Schema Version attribute, and that claim full conformance to this specification, MAY use the use the value “1.0.2011.11” for that attribute.