Extension and Modifications to the OASIS Energy Interop Operation Schema for OpenADR 2.0

July 26, 2013

Introduction
The OpenADR 2.0 specification is a profile of the OASIS Energy Interop (EI) version 1.0 specification and schema (EI). This document details the extensions and modifications made to the OASIS EI Schema by the OpenADR Alliance (OADRA) to create their profiles. This is in reference to EI schema version number 1.0. The main purpose of this document is to enable the EI Technical Committee to incorporate appropriate changes from the OADRA and align future versions of EI with the OADRA profiles.

The approach taken by the OADR A was to keep changes to a minimum and only make changes that were deemed to be necessary, as requested by the demand response market implementations and vendors.

In general the OADRA created a set of schemas for their profiles that were derived from the EI schemas in the following ways:

1. Modifications to facilitate payload exchanges
2. Sub-setting of and restrictions to existing EI schemas
3. Proper extensions to add attributes to existing EI schemas
4. Modifications to the EI schema to fix shortcomings or bugs
5. Addition of transport mechanisms to support testing and certifications
6. Addition of security architecture and implementations
7. Development of conformance statements for interoperability

Only in the case #4 above were changes made to the existing EI schema that may be deemed to be incompatible and these changes were kept to an absolute minimum.

Namespaces
The OADRA created separate namespaces from the EI schema. This was done for a variety of reasons, but all the attributes and elements of the EI schema were maintained such that any XML file created with the OADRA schema files would validate against the EI schemas with the possible exceptions of the extensions and modifications that are outlined in the remainder of this document.

Authored by Ed Koch, Honeywell and Jim Zuber, QualityLogic
The OADRA currently has an A (restricted devices) and B (fully functional endpoints) profile. While the A profile is a proper subset of the B profile with respect to the features and attributes of the schema they are each in separate namespaces.

The general naming convention used by the OADRA is that elements and attributes from EI maintain the same ei prefixes and names whereas new attributes and elements defined by the OADRA have an oadr prefix. Therefore any schema elements show in this document with the “oadr” prefix are from the OADRA and any with the “ei” prefix are from the original EI schema.

### Payloads and Interaction Patterns

This section covers the interactions patterns of each of the services and the extensions that were made to accommodate the exchange of payloads. The details around changes to the major data elements of the payloads (e.g. eiEvent) are covered in a subsequent section. Two OADRA feature sets were created to support these services, OpenADR 2.0a Profile, and OpenADR 2.0b Profile.

The 2.0a profile was designed for low end restricted functionality VEN’s and only contains the features deemed absolutely necessary for a low end device and DR markets that want to send simple signals. The following services are supported by a 2.0a profile VEN:

- Event service (basic)

The B profile was designed for more fully functional VEN’s and contains a superset of the A profile features, to support advanced DR retail and wholesale markets. The following services are supported by a B profile VEN:

- Registration service
- Event service (basic and advanced)
- Opt service
- Report service
- Poll service

Each of these services is covered in more detail below with respect to the payloads that are used.

### General oadrPayloads

For security purposes, a general oadrPayload wrapper was created to allow the signing of any payload exchanged. This new payload wrapper did not entail the modification of the existing EI schema and is shown below.
The oadrSignedObject is simply a switch that may contain any one of the service specific payloads that are described in each of the sections below.

In addition the oadrResponse payload was created as a general purpose response used in many of the interactions for the services described below. It is defined as:
As can be seen this is a simple payload that is constructed of a subset of the eiResponse element and a venID.

**Event Service**

The interaction patterns for the event service have been simplified from EI in order to support both a PUSH and a PULL model of interactions and to use a common payload between the VTN and VEN for the exchange of those payloads. Thus the oadrDistributeEvent payload is the same whether the VTN is pushing events to the VEN or the VEN is requesting them.

The Event service consists of the following operations and payloads:

<table>
<thead>
<tr>
<th>Request Payload</th>
<th>Response Payload</th>
<th>Requestor</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>oadrRequestEvent</td>
<td>oadrDistributeEvent</td>
<td>VEN</td>
<td>VTN</td>
</tr>
<tr>
<td>oadrDistributeEvent (PUSH service)</td>
<td></td>
<td>VTN</td>
<td>VEN</td>
</tr>
<tr>
<td>oadrCreatedEvent</td>
<td>oadrResponse</td>
<td>VEN</td>
<td>VTN</td>
</tr>
</tbody>
</table>
**oadrRequestEvent**

The oadrRequestEvent payload is simply a wrapper around the existing eiRequestEvent payload as shown below. Note that the oadrRequestEvent always returns all the pending events (replyLimit notwithstanding) and thus the pending Events operation is not necessary in OADRA.
The oadrDistributeEvent payload is shown below and is a simple extension of the eiDistribute event payload. The new attributes are the following:

- **eiResponse** attribute added so that the same oadrDistribute payload could be used for both PUSH and PULL scenarios.
- **oadrResponseRequired** - used to signify to the VEN if a response to this message is required by the VTN. This allows “broadcast” use cases to be supported which do not required responses from the VEN.
The oadrCreatedEvent payload is as follows.
As can be seen it is simply a wrapper around the existing eiCreatedEvent payload.

**Opt Service**

The Opt service consists of the following operations and payloads

<table>
<thead>
<tr>
<th>Request Payload</th>
<th>Response Payload</th>
<th>Requestor</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>oadrCreateOpt</code></td>
<td><code>oadrCreatedOpt</code></td>
<td>VEN</td>
<td>VTN</td>
</tr>
<tr>
<td><code>oadrCancelOpt</code></td>
<td><code>oadrCanceledOpt</code></td>
<td>VEN</td>
<td>VTN</td>
</tr>
</tbody>
</table>

**oadrCreateOpt**

As shown below the `oadrCreateOpt` payload extends the `eiCreateOpt` payload with the following attributes:

- `requestID` – for tracking transactions
- `qualifiedEventID` – for specifying a specific event for which to opt out
- `eiTarget` – for specifying a specific set of resources to apply the opt out to
- `oadrDeviceClass` – for specifying a set of devices that may opt out
Note that all the above attributes simply reuse existing elements in the EI schema, but add them to the oadrCreateOpt payload.

Both the eiTarget and oadrDeviceClass could be used to define the scope of the availability schedule as well as defining the scope of optIn/optOut, depending on the usage of oadrCreateOpt.

The reason OADRA created a separate oadrDevice class element is to allow the endDeviceAsset subelement in eiTarget to be used as originally intended mrid values, where as in oadrDevice class we have specific set of enumerated device class values defined in the conformance rules. So, a VEN might list many mrid values in eiTarget and pool pump in oadrDeviceClass and availability schedule would apply to the ANDd subset. This is consistent with the other two places in the schema where we used both eiTarget and device classes (reports and events)
Generated by XMLSpy
As shown below the oadrCreatedOpt and the eiCreatedOpt are the same.
As shown below the oadrCancelOpt payload made a simple extension to eiCancelOpt to add a requestID.
**oadrCancledOpt**

As shown below there are no difference between oadrCancledOpt and eiCancledOpt.
Registration Service
The registration service is used to allow a VEN to register with a VTN.

The registration service consists of the following operations and payloads.

<table>
<thead>
<tr>
<th>Request Payload</th>
<th>Response Payload</th>
<th>Requestor</th>
<th>Responder</th>
</tr>
</thead>
</table>

Authorized by Ed Koch, Honeywell and Jim Zuber, QualityLogic
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>VEN</th>
<th>VTN</th>
</tr>
</thead>
<tbody>
<tr>
<td>oadrCreatePartyRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrCreatedPartyRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrQueryRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrCreatedPartyRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrCancelPartyRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrCanceledPartyRegistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrRequestReregistration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oadrResponse</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**oadrCreatePartyRegistration**

In EI there was very little defined concerning the attributes necessary to support registration so as shown below there were a number of attributes added as deemed necessary by OADRA. The general rule the OADRA applied to determining which attributes to add were to restrict it only to those that were deemed absolutely necessary to establish communications between a VTN and VEN. Thus the attributes reflect things that help the VTN establish communications and identify the VEN.
**oadrCreatePartyRegistrationType**

- **pyIDrequestID**
  - Type: xs:string
  - A ID used to match up a logical transaction request and response.

- **setregistrationID**
  - Type: xs:string
  - Used for re-registering an existing registration.

- **setvenID**
  - Type: xs:string
  - Not used.

- **oadrProfileName**
  - Type: oadr:oadrProfileName
  - OpenADR profile name such as 2.0a or 2.0b.

- **oadrTransportName**
  - Type: oadr:oadrTransportType
  - OpenADR transport name such as simple-http or smpp.

- **oadrTransportAddress**
  - Type: xs:string
  - Address of this VEN. Not required if http pull model.

- **oadrReportOnly**
  - Type: xs:boolean
  - ReportOnlyDeviceFlag - True or False.

- **oadrXmlSignature**
  - Type: xs:boolean
  - Implementation supports XML signatures - True or False.

- **oadrVendorName**
  - Type: xs:string
  - Human readable name for VEN.

- **oadrHttpPullModel**
  - Type: xs:boolean
  - If transport is simple-http indicates if VEN is operating in pull-exchange model - true or false.

**Generated by XMLSpy**  www.altova.com

**Authored by Ed Koch, Honeywell and Jim Zuber, QualityLogic**
oadrCreatedPartyRegistration
In EI the eiCreatedPartyRegistration was designed as a simple response to the eiCreatePartyRegistration operation. In OADRA this payload was extended and serves as a response to both the oadrCreatePartyRegistration and the new oadrQueryPartyRegistration. As shown below it contains extra attributes that can be used to further control the interactions and communications between the VEN and the VTN.
oadrQueryRegistration
This operation does not exist in EI and serves as a way for the VEN to query the VTN about certain attributes that may be necessary for any VEN to communicate with the VTN.

oadrCancelPartyRegistration
This operation can be initiated by either the VEN or VTN. As shown below it differs slightly from the EI version in that the EI version uses registreePartyID and registrarPartyID, wherein the OADRA relies upon the registrationID and optionally the venID.
oadrCanceledPartyRegistration

This operation is a response to the oadrCanceledPartyRegistration and likewise, as shown below, it differs slightly from the EI version in that the EI version uses respondingPartyID, registreePartyID and registrarPartyID, wherein the OADRA relies upon the registrationID and optionally the venID.
oadrRequestReregistration
This payload is semantically similar to the eiRequestPartyRegistration although as shown below it is much simpler in what attributes it requires.
Report Service

The reporting service defined in OADRA is a simplified version of that defined in EI. OpenADR 2.0 does not use the notion of a historian. Instead the approach used is that the generator of the report (VEN or VTN) uses the oadrRegisterReport operation to notify the other party (i.e. the recipient of the report) of its reporting capabilities and if the other party wishes to receive one of those reports it then uses the oadrCreateReport operation to have the report generated. The generator of the report then uses the oadrUpdateReport to send the report to the recipient and likewise if it is a periodic report then the recipient can use the oadrCancelReport to cease the report being generated and sent. All report types whether they are periodic, one time, projections, historical, etc. all use the oadrUpdateReport operation to send the report.

The report service consists of the following operations and payloads.

<table>
<thead>
<tr>
<th>Request Payload</th>
<th>Response Payload</th>
<th>Requestor</th>
<th>Responder</th>
</tr>
</thead>
</table>

Authored by Ed Koch, Honeywell and Jim Zuber, QualityLogic
<table>
<thead>
<tr>
<th>oadrRegisterReport</th>
<th>oadrRegisteredReport</th>
<th>VEN or VTN</th>
<th>VTN or VEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>oadrCreateReport</td>
<td>oadrCreatedReport</td>
<td>VEN or VTN</td>
<td>VTN or VEN</td>
</tr>
<tr>
<td>oadrUpdateReport</td>
<td>oadrUpdatedReport</td>
<td>VEN or VTN</td>
<td>VTN or VEN</td>
</tr>
<tr>
<td>oadrCancelReport</td>
<td>oadrCanceledReport</td>
<td>VEN or VTN</td>
<td>VTN or VEN</td>
</tr>
</tbody>
</table>

**oadrUpdateReport**

As can be seen below the oadrUpdateReport is a very simple extension of eiUpdateReport in that it also includes a venID. The oadrReport element is an extension to the eiReport element as will be discussed in a subsequent section. The oadrUpdateReport is used to send all reports in OADRA and not just periodic changes as in EI.
**oadrUpdatedReport**
As can be seen below oadrUpdatedReport is a simple extension to eiUpdatedReport. It adds a venID attribute and the oadrCancelReport element that allows for reports to be cancelled as part of this transaction.
**oadrCancelReport**

As shown below the oadrCancelReport is a simple extension/subset of the eiCancelReport.
As shown below the oadrCanceledReport is a simple extension/subset of the eiCanceledReport. Note that in OADRA reportRequestID is the unique identifier of a report which is why reportedID was omitted.
oadrRegisterReport

This operation does not exist in EI. As noted above it is intended that one party use this to notify the other party what its reporting capabilities are. As shown below it looks almost identical to the oadrUpdateReport payload in that the primary data being transported is an oadrReport (i.e. eiReport). By design a special type of report called the METADATA report is used to portray the type of reports that
can be sent and MEATADATA reports have all the same attributes that are defined in oadrReport (i.e. eiReport).

**oadrRegisteredReport**
This simple payload is a response to the oadrRegisterReport payload and is shown below. One thing worth noting is the oadrReportRequest attribute which allows the report recipient to request a report as part of the acknowledgement of the report registration by the report generator.
oadrCreateReport
As shown below oadrCreateReport is a simple subset of eiCreateReport. As can be seen below there are attributes in eiCreateReport and eiReportRequest elements that were mandatory in EI but eliminated in OADRA.

Generated by XMLSpy  www.altova.com
oadrCreatedReport

As shown below oadrCreatedReport is a simple extension to eiCreatedReport.

Authored by Ed Koch, Honeywell and Jim Zuber, QualityLogic
**oadrPoll**

The oadrPoll service is a general service used to simulate a PUSH interaction that is initiated by the VEN. This service did not exist in EI.

It consists of the following payloads.

<table>
<thead>
<tr>
<th>Request Payload</th>
<th>Response Payload</th>
<th>Requestor</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>oadrPoll</td>
<td>One of the following:</td>
<td>VEN</td>
<td>VTN</td>
</tr>
<tr>
<td></td>
<td>• oadrResponse</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrDistributeEvent</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrCreateReport</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrRegisterReport</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrCancelReport</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrUpdateReport</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrCancelPartyRegistration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• oadrRequestReregistration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The response payloads are defined as above.

**Sub-setting and Restrictions**

Many of the elements and attributes in the existing EI schema are optional and not used by OpenADR 2.0. Such elements do not appear in the OpenADR 2.0 schemas and thus do not show up in the diagrams depicted throughout this document. Furthermore all elements that were required in the EI schema should appear in the OpenADR schema unless they are specifically called out as a modification in this document.

Furthermore there may be elements that were optional in EI, but made mandatory in OADRA.

Authored by Ed Koch, Honeywell and Jim Zuber, QualityLogic
Therefore many of the apparent differences between the OpenADR 2.0 schema and the EI schemas are nothing more than optional elements of EI either not being used or made mandatory in OADRA and thus are not changes or modifications to the existing EI schemas.

Differences between EI and OADRA of his nature are not explicitly called out in this document because they did not represent any changes to the EI schema. Those interested in what these differences are should read the OpenADR profile specifications.

**Extensions and Modifications**

This section details the extensions and modifications that were made to the EI schema. Changes of this nature fall into three general categories:

1. Extensions made to the EI schema using predefined EI extension mechanisms and thus are extensions that exist in the EI namespace.
2. Extensions made to the EI schema by adding new attributes to existing EI elements and creating oadr elements in the oadr namespace.
3. Outright modifications to the existing EI schema

Note that category one above does not require any changes to the existing EI schema’s while categories two and three do. As previously noted changes of type two and three above where kept to a minimum.

**General Changes**

**EIEvent**

As shown below eiEvent in OADRA is a proper subset of eiEvent in EI.

OADRA version
EI version
For more details on which elements within OADRA are required and which are not used then the OADRA schema’s may be consulted, but in all cases no modifications were made to the existing EI schemas for eiEvent.

Of special note are the extensions that were made to SignalTypeEnumeratedType to add the following signal types using the supported “x-“ method of extending enumerated types.

- x-loadControlCapacity
- x-loadControlLevelOffset
- x-loadControlPercentOffset
- x-loadControlSetpoint
EIReport
As shown below, at the highest level the oadrReport is compatible with eiReport with the exception of the oadrReportDescription attribute versus the eiReportDescription attribute. These are shown in more detail below. Note that the reportName attribute designates the report type and currently supports the following standard OADRA standard reports:

METADATA_HISTORY_USAGE
HISTORY_USAGE
METADATA_HISTORY_GREENBUTTON
HISTORY_GREENBUTTON
METADATA_TELEMETRY_USAGE
TELEMETRY_USAGE
METADATA_TELEMETRY_STATUS
TELEMETRY_STATUS

This does not preclude the addition of other user defined report types.
As shown below the two schema are compatible with the following extensions being added:

- new data types added to the itemBase using the established substitution group mechanics. The new types are prefixed with “oadr”
- New oadrSamplingRate attribute added to ReportDescription
Also of special note are the eiReport::strm:intervals that have been extended using the stream substitutions groups in the manner shown below. Note the addition of the oadrReportPayload and the oadrGBP Payload. Of special note is that in OADRA each report interval may have more than one data point, while the current EI schema cardinality for streamPayloadBase is 1. The OADR thus changed the cardinality of streamPayloadBase to 1 to many.

The oadrReportPayload is an extension to the reportPayload in EI as shown below. A new attribute named oadrDataQuality has been added an in addition a payload type called oadrPayloadResourceStatus has been added using the established substitution groups.
Furthermore, as can be seen in oadReport above, the oadrGBPayload was also added to the StreamPayloadBase substitution group. This is intended to hold GreenButton payloads and is defined as shown below.