ISO New England Wind Integration Data Exchange Specification

This document is the data exchange specification for the ISO-NE Wind Integration Phase 1 project using the **e-terra***renewableplan* product. This document is designed to assist ISO New England (ISO-NE) Wind Forecast Providers and Wind Plant Lead Participants in the development of their web services interface to the **e-terra***renewableplan* web service application

Date:	**July 27, 2016**
Version:	**12.1**

Copyright and Proprietary Information

Trademarks

Contents

About This Document

This document describes the SOAP messages and the Authentication and Authorization process used to exchange forecast, narrative, and telemetry data between Forecast Providers and Wind Plant Lead Participants through web service interfaces. This document explains how to access the wind integration web service, presents the format and construction of SOAP messages used to exchange data, and briefly describes the Authentication and Authorization methods used to ensure security.

Purpose of this Document

This guide is designed to assist Forecast Providers and Wind Plant Lead Participants to develop interfaces that interact and exchange forecast, narrative and telemetry data using the ISO-NE Wind Integration web services. This document will help Wind Integration Participants comprehend and construct the appropriate data messages essential for data exchange with the ISO-NE Wind Integration web services.

This guide describes all the submit/query messages that can be constructed in order to exchange data between both Wind Plant Lead Participants and Forecast Providers and the ISO-NE **e-terra***renewableplan* application. This guide specifies required and/or optional data in messages from Wind Integration Participants and describes the data messages that can be requested from Wind Integration web service application. This guide also describes the purpose and construction of SOAP messages, the adopted format for data exchange between all Wind Integration Participants and Wind Integration web service application.

Scope and Prerequisite Knowledge

This document is intended to be used by all ISO New England Wind Integration Participants involved in wind integration as an aid in developing new interfaces.

Users of this Guide should be familiar with Extensible Markup Language (XML), Web Services, HTTP/HTTPS protocols, ISO New England's governing documents, business rules and operating procedures as well as have a working understanding of how the energy market works and functions in New England. Refer to the *References and Additional Information* section for helpful links.

Structure of this Document

- Chapter 1 gives an overview of the Wind Integration web service including Wind Integration web service design and access, participant roles, and authentication and authorization
- Chapter 2 describes SOAP messages; from their construction, to the format they are displayed in this document

- Chapter 3 describes restrictions and validations put on data submitted to Wind Integration
- Chapters 4 describes the Forecaster Web Services
- Chapter 5 describes the Wind Plant Lead Participant Web Services

References and Additional Information

Additional information about Extensible Markup Language (XML), Web Services, and other helpful information can be found at the following links:

- XML –
  - **http://www.w3.org** > XML Technology
  - **http://www.w3schools.com** > Learn XML
- Web Services –
  - **http://www.w3.org** > Web Service Technology
  - **http://www.w3schools.com** > Learn Web Services
- SOAP –
  - **http://www.w3schools.com** > Learn SOAP

ISO New England governing documents include the Transmission, Markets & Services Tariff, ISO New England Manuals and Operating Procedures. In particular, Operation Procedures 14 and 18 deals specifically with wind communications. This documentation can be found at the following location:

- **http://www.iso-ne.com > Rules & Procedures**

Change Summary

Rev	Trans #	Date	Comments
1.0	908E1798.1	02/27/10	First released version transmitted to ISONE.
2.0	908E1798.2	03/26/12	Second version changes reflect comments created by ISONE, more generic data types, and endpoints that are based on the client.
3.0	908E1798.3	05/02/12	This version (v3.0) incorporates feedback from ISONE and the following new functionality and technical revisions: • Ramp Event web service • XML namespace naming convention change • Operation-specific Faults (i.e., ForecastFault) compared to a single generic Fault
4.0	908E1798.4	06/07/12	This version incorporates: • The ability to "cancel" Ramp Events by submitting empty data points • Specifies that UTC DateTime will be returned for all date and time values retrieved from the database
5.0	908E1798.5	06/15/12	This version incorporates: • Specifying WPFA query operation for Forecaster, removing Daily and Hourly WPFA operations • Adding querying for WPFA for Wind Plant Lead Participants
6.0	908E1798.6	06/28/12	Changes include designated "Distributed Generation" as slated for development in a future release.
7.0	908E1798.7	08/01/12	Provided list of available authorization roles
8.0	908E1798.8	08/13/12	Replaced "Designated Entity" and variations with the corresponding "Lead Participant" variation.
9.0	908E1798.9	09/07/12	Updates to better align with the WSDL files along with general corrections.
10.0	908E1798.10	09/12/12	Minor updates to better support participant guide merging.
11.0	908E1798.11	11/01/12	Minor updates to align with WSDL updates.
12.0	908E1798.12	02/10/16	Updated to include references to new element AssetIdentifier (NE32205). Fixed the bug that was causing terminated user name (i.e. old, changed name) to appear in the log together with the new name (NE32181)
12.1		07/27/16	Changes for Asset ID enhancement.

1. Web Service Overview

This chapter gives an overview of web service concepts associated with the Wind Integration web service, how to access the web services, and the different roles.

1.1 Web Service Design

The **e-terra***renewableplan* web service, or programmatic interface, is described by operations that are defined in the **e-terra***renewableplan* WSDL files. These operations also describe the messages that are used to transfer data between a participating parties and the **e-terra***renewableplan* application through the interaction between participating client interfaces and the **e-terra***renewableplan* web service. The web service operation messages use SOAP format for data transmission, which is discussed in more detail in chapter 2 entitled SOAP Messages.

All web service operations follow a request/reply pattern that is typical of HTTP(S) communication. A request may contain a message that modifies (or submits) data or it may contain a message that queries for data. A reply contains a message that is either: 1) a confirmation of data modification 2) an error (i.e., “fault”) or 3) the response to a query containing the results.

Any web service operation that allows data to be modified will have a corresponding web service query operation. In many cases, this web service operation pairing will have a submit message that contains the exact same data as a query response message. This relationship is referred to as having symmetrical messages. However, not all submit/query response messages are symmetrical. There are cases in which more data is returned in a query response message than can be contained in a submittal message (e.g., messages that contain telemetry data).

There are some web service operations that simply have query messages, and they are used for the sole purpose of requesting specific data from the **e-terra***renewableplan* application. A more in depth description of the symmetry of web service operation messages used by the Wind Integration web service can be found in section 2.6 Submittal and Query Response Symmetry.

The WSDL and the associated XML Schema Definition (XSD) file do not enforce the number of XML entities that are expected in any type of submission. For example, the WSDL operations provide the ability to submit forecasts but do not validate submittals for the appropriate number of forecasts at the expected time intervals; this type of validation occurs in the application layer. Additionally, a forecast submission that is technically valid according to the WSDL may not include all of the required data points. Missing Wind Plant forecast data will be monitored by the **e-terra***renewableplan* application and the appropriate alarm will be raised.

1.2 Accessing the Wind Integration Web Service

The **e-terra***renewableplan* web services are based on SOAP (version 1.1) over HTTPS (version 1.1). A client application that accesses the application can be written in nearly any modern enterprise technology and language, such as Java, .NET, C++, Ruby, PHP, etc. The web service is accessible to authorized participants through an ISO New England published URL, and all submittals and queries are serviced via the same URL.

Authentication to the interface uses digital certificates issued by ISO New England. Potential users need to register with the Customer Support department within ISO New England in order to obtain valid certification for access to the programmatic interface. Once the company is registered with ISO-NE, the Participant's company appoints a Security Administrator (a member of their own staff) who in turn grants permissions (or roles) to the potential users of the Wind Integration web service application or any other market application from that participant company. Users contact their own Security Administrator for roles to be opened on their certificates (which reside on their own computers). Refer to the **Customer Asset Management System Application Group Roles** document on the ISO web page for **User Guides**.

The **e-terra***renewableplan* application will implement an already well understood Authentication and Authorization (A&A) architecture as well as the supporting processes that are already in place at ISO New England – that of the Market User Interface (MUI). Essentially, the **e-terra***renewableplan* will implement this architecture to create users, to manage users, to associate those users with defined roles, to perform user authentication, and to authorize user access to web service operations. A detailed design document for this architecture already exists; for more information on that architecture, please review the "*Market User Interface (MUI) Authentication and Authorization Delta Design Note"*.

In addition to adopting the aforementioned architecture, the **e-terra***renewableplan* application will also be extending its authorization capabilities. Specifically, the **e-terra***renewableplan* application's schema will include additional tables that define relationships between participants and entities (i.e., a wind plant, meteorological "met" measurement, etc.) as well as between participants and different schedule types (i.e., forecasts, telemetry values, narratives, etc.). The **e-terra***renewableplan* application will use relationships defined in these tables to ensure that participants are authorized to access specific entities and schedules and to establish their permissions (i.e., read only, read/write) for each entity.

These relationships discussed above will utilize the A&A-provided PARTICIPANTID. Consequently, the processes by which participants are created and maintained will need to account for these additional authorization tables. For additional information regarding the design of these permission tables, please review the "*Wind Integration Phase 1 DDN*" document.

Questions or inquires about certifications for **e-terra***renewableplan* application access should be addressed to the Customer Support Department at ISO New England.

1.3 Participant Roles

Roles define the web service operations/messages that an **e-terra***renewableplan* user can use to submit or query data. Distinct roles are associated with different web service operations (i.e., Forecast, Telemetry, Narrative, Ramp Event, etc.). Roles are also divided into two sub-types; "Read Only" and "Read/Write". "Read Only" roles restrict the user to Query messages associated with their role's web service operations, while "Read/Write" roles allow the use of both Submit and Query messages.

For example, a Forecast "Read/Write" role assigned to a user authorizes that user to invoke both the "QueryForecast" and "SubmitForecast" web service operations.

Following the conventions listed above, the following roles are defined:

- WPLPReadOnly
- WPLPReadWrite

1.4 Web Service and XML Schema Definition Files

The ISONE Wind Integration Web Services depends on three different files that are referenced throughout this document. The following table presents the list of these files, their namespaces, and a brief description.

Filename	Namespace	Description
WindPlantLeadParticipantService.wsdl	urn:com.alstom.isone.windint.wplp:1-0:wsdl	The Web Service definition describing the operations available to a "WPLP" and how they should be accessed. This web service file references only the XML elements contained in WindIntegration.xsd.

Filename	Namespace	Description
WindIntegration.xsd	urn:com.alstom.isone.windint.windintegration:1-0	Contains the elements specific to the ISONE wind integration effort. The elements contained in top section of this file are referenced in each of the WSDL files. The elements defined in the Definition sections define simple and complex data types as explained in the table below

The following table describes the elements and types defined in the Definition sections of the WindIntegration.xsd file. The Definition sections help organize the elements and types according to function and facilitate discussion in this document. The Definition sections are delimited by comments in the WindIntegration.xsd file.

Definition Section	Description
CommonObjects	As discussed in 3.1.1.2, contains reusable (i.e., referenced at least twice) data type definitions.
CommonOperations	As discussed in 3.1.1.3, contains data type definitions that present the available Category, Schedule, Entities and associated permissions of the requestor.
PowerSchedule	Contains the data types for querying and submitting schedules related to power generation. This section is used to build the forecast-related querying and submittal elements. The data types contained within this section satisfy the requirements to create the structures necessary for the these forecasts: • Short Term Wind Plant Forecast • Medium Term Wind Plant Forecast • Long Term Wind Plant Forecast • Hourly Wind Plant Future Availability • Daily Wind Plant Future Availability

2. SOAP Messages

This chapter describes SOAP messages and how they are used in the **e-terra***renewableplan* application. This chapter describes constructing SOAP messages, restrictions on data submitted, and SOAP format/documentation in this guide.

SOAP is a specification for exchanging information involving Web Services. SOAP messages are constructed using Extensible Markup Language (XML) as a structure to store data. This XML structure is wrapped in a SOAP envelope that carries processing instructions and descriptions of the data for interpretation by an interface or Web Service. For reference, these operations are all described in Web Services Definition Language (WSDL) files (see section 1.4).

2.1 Submit and Query Responses

Each message sent is an “all or nothing” event. The **e-terra***renewableplan* application will use database transactions, such that a commit will only occur on successful processing of an entire SOAP envelope. If an exception occurs while processing a message, a fault will be sent to the user with the appropriate error messages. Specifically, querying will not return any results and submittal transactions will be saved, however, they will be marked as invalid by the **e-terra***renewableplan* application.

```
<ForecastFault>
  <!-- One or more repetitions -->
  <error>
    <message>?</message>
    <number>?</number>
  </error>
</ForecastFault>
```

Query messages return a wide variety of information, and as such they do not have a standard response message unless the message returns a fault similar to the one shown above. As the example above illustrates, the faults are named according to the type of operation invoked; however, each of these are merely instances of the same Fault element as is defined in the “*WindIntegration.xsd*” file. A description of each fault is contained within the web services sections 4 and 5; specifically, the first element presented in each the “Data Returned’ sub-sections.

Note that, similar to submit operations, query messages that contain invalid data are also treated as “all or nothing” events; invalid query response will return a fault.

Each Submit message that is sent to the **e-terra***renewableplan* application has a standard response message that confirms the message was received and processed. The response message contains a transaction ID that is used to track/indicate the confirmation of the message submitted. The standard response message to a Submit message is shown below

Standard Submit Response:

```
<soap:Envelope
     xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <Submit?Response
           xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <SubmitStatusResponse>
            <success>
               <!-- Only 1 repetition -->
               <transactionId>?</transactionId>
            </success>
         </SubmitStatusResponse>
      </Submit?Response>
   </soap:Body>
</soap:Envelope>
```

2.2 Format and Construction

SOAP messages are XML formatted structures wrapped in a SOAP envelope. XML formatted messages are organized with elements and attributes, and the structure looks very similar to HTML formatted messages. A simple XML message is shown below:

XML Message

```
<note>
  <to>Mike</to>
  <from>John</from>
  <heading>Reminder</heading>
  <body>Don't forget fishing this weekend!</body>
</note>
```

Messages used in the **e-terra***renewableplan* application look similar to the example above, however the XML formatted message is wrapped in a SOAP envelope. This SOAP envelope is shown below:

SOAP Envelope

```
<soapenv:Envelope
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
  <soapenv:Header/>
  <soapenv:Body>
    ...
    (XML Message here)
    ...
  </soapenv:Body>
</soapenv:Envelope>
```

An XML formatted message and a SOAP envelope[1] come together to form a SOAP message that is used to exchange data between a third party interface and a Web Service. With respect to the **e-terra***renewableplan* application, client interfaces are constructed and operated by participating wind plant operators (i.e., Wind Plant Lead Participants) and forecasters, while the Web Service is operated by ISO New England. An example of a complete SOAP message, an XML formatted message with a SOAP envelope, that is documented in this guide is shown below:

Full SOAP Format

```
<soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryForecast>
         <wint:ScheduleRequest>
            <wint:Schedule>
               <wint:identifier>?</wint:identifier>
            </wint:Schedule>
            <wint:TimeRange>
               <wint:fromTime>?</wint:fromTime>
               <wint:toTime>?</wint:toTime>
            </wint:TimeRange>
            <wint:Entities>
               <wint:Entity>
                  <wint:identifier>?</wint:identifier>
               </wint:Entity>
            </wint:Entities>
         </wint:ScheduleRequest>
      </wint:QueryForecast>
   </soapenv:Body>
</soapenv:Envelope>
```

---

[1] The samples within this document reference SOAP 1.1 as indicated by the namespace "`http://schemas.xmlsoap.org/soap/envelope/`".

In examples like the one shown above, question marks indicate elements or attributes that should be populated with data. The `wint:` prefix that appears in each element of the XML body denotes an XML namespace that the **e-terra***renewableplan* application uses to avoid **naming** conflicts.

The example message above demonstrates the format of messages that exists throughout this document. This format is used because it shows the complete SOAP message template used to exchange data with the **e-terra***renewableplan* application web services. Each of the SOAP messages in the "Full SOAP Format" section of the web service operations can be used to submit/query data from the **e-terra***renewableplan* application, as long as the elements and attributes contain valid values. This document uses a color convention for distinguishing the various parts of the SOAP message. The color convention for items within these messages is shown below:

Element	Element Value	Attribute	Attribute Value	Comment
(brown)	(black)	(magenta)	(blue)	(green)

The **e-terra***renewableplan* application data that is contained within the SOAP body is defined in the WSDL and XSD files included in section 1.4. Within this document, the individual XML elements and attributes that comprise the **e-terra***renewableplan* application data are described in subsequent chapters and using a table format with the following column headings:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments

Opt. – Indicates whether an element or attribute is optional

Nil. – Indicates whether an element or attribute is nillable

Element or Attribute – The name of element or attribute

Data Type; Format – Specifies the data type and format for the data

Comments – Specific information about the element or attribute

2.3 Optionality and Nillability

Some messages contain elements and/or attributes that are optional. Element and group optionality is indicated in the XML Schema by specifying `minOccurs=”0”`. For element attributes, `use=”optional”` indicates optionality.

```
<complexType name="CategoryIdentityType">
  <sequence>
    <element name="identifier" type="string"
             minOccurs="1" maxOccurs="1"/>
    <element name="name" type="string" minOccurs="0" maxOccurs="1"/>
    <element name="description" type="string"
             minOccurs="0" maxOccurs="1"/>
  </sequence>
</complexType>
```

The next sections describe the conventions for handling optional attributes and elements that can be null. Note that we currently have no optional attributes or nillable elements.

2.3.1 Submit Messages with Nillable Elements

Any element that is marked as `nil=”true”` will be interpreted as meaning "set the value in the database to NULL". The NULL value will be effective according to standard effective dating. A large number of Submit type messages contain elements that can be nil.

2.3.2 Query Response Messages with Optional Attributes

Any Query Response message that contains optional attributes will have values for that attribute in the XML if the database has corresponding values. If the database does not have a corresponding value, the attribute tag will not appear in the XML.

2.3.3 Query Response Messages with Nillable Elements

Any Query Response message that contains nillable elements will specify `nil=”true”` in the XML element if the database has a NULL value for that element.

2.4 Query Filters

When querying data from the **e-terra***renewableplan* application, a Wind Integration Participant will submit a query request message that contains filters to limit the data that is queried. Typical query filters include the Schedule identifier, the TimeRange, and the Entity identifier (i.e., Wind Plant, System, DispatchZone, Met Measurement, etc.), though other filter criteria are possible depending on the nature of the data being queried (i.e., Category identifier, etc.). An example of a query message is shown below; the bold text identifies the query filter elements:

Full SOAP Message

```
<soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryForecast>
         <wint:ScheduleRequest>
            <wint:Schedule>
               <wint:identifier>?</wint:identifier>
            </wint:Schedule>
            <wint:TimeRange>
               <wint:fromTime>?</wint:fromTime>
               <wint:toTime>?</wint:toTime>
            </wint:TimeRange>
            <wint:Entities>
               <wint:Entity>
                  <wint:identifier>?</wint:identifier>
               </wint:Entity>
            </wint:Entities>
         </wint:ScheduleRequest>
      </wint:QueryForecast>
   </soapenv:Body>
</soapenv:Envelope>
```

In this example, the `Schedule` and `TimeRange` are required elements, meaning that both must be included in the request. In addition, the `Entity` and `Category` (not shown) elements are optional – which are intended to further narrow the amount of data returned.

2.5 Sample SOAP Message Format

The "Sample of Submittal (or Query)" message that follows each "Full SOAP Format" message in this document uses the full SOAP message, however the elements/attributes have made up values to show how data could be submitted. These sample messages also have the SOAP envelope, namespaces, comments, and party attributes removed in order reduce the overall length of this document. The samples below show the differences between a full SOAP message and its corresponding sample message. The full SOAP message below has the elements and attributes that are removed in its corresponding sample message highlighted in white.

```
<soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryForecast>
         <wint:ScheduleRequest>
            <wint:Schedule>
               <wint:identifier>?</wint:identifier>
            </wint:Schedule>
            <wint:TimeRange>
               <wint:fromTime>?</wint:fromTime>
               <wint:toTime>?</wint:toTime>
            </wint:TimeRange>
            <wint:Entities>
               <wint:Entity>
                  <wint:identifier>?</wint:identifier>
               </wint:Entity>
            </wint:Entities>
         </wint:ScheduleRequest>
      </wint:QueryForecast>
   </soapenv:Body>
</soapenv:Envelope>
```

This full SOAP message would have this sample message documented below in a simplified format.

```
<QueryForecast>
   <ScheduleRequest>
      <Schedule>
         <identifier>?</identifier>
      </Schedule>
      <TimeRange>
         <fromTime>?</fromTime>
         <toTime>?</toTime>
      </TimeRange>
      <Entities>
         <Entity>
            <identifier>?</identifier>
         </Entity>
      </Entities>
   </ScheduleRequest>
</QueryForecast>
```

The SOAP envelope, namespaces, comments, and attributes that were removed in this example are common throughout this document; they will always be removed in sample messages. The sample message above will not process without the highlighted portions of the full SOAP message, with the exception of the lines of comments (indicated in green). Lines of comments are inserted in the full SOAP message to indicate optionality and element repetition, and a message will process with or without them. Note that the above example is taken from the Query message in the `QueryForecast` operation of the Forecaster Web Service (see section 4.0).

Sample messages may or may not have the elements/attributes that are optional included for the common reason of saving document length. It is important to note that a message will process with or without optional elements and attributes included, even though optional elements/attributes may not be shown in a sample SOAP message.

2.6 Submittal and Query Response Symmetry

In general, the XML structure of the submitted data is almost identical to the Query response message. Specifically, optional elements used strictly for informational purposes (i.e., “name”) will be included in the query response. These same elements, however, will be ignored if included in a Query/Submit request. Ignored elements are excluded from the tables that define the inputs and outputs for the Web services.

The XML below shows sample data for a Submit message for a forecast.

```xml
<SubmitForecast>
    <CreateSchedule>
        <Schedule>
            <identifier>1</identifier>
        </Schedule>
        <TimeRange>
            <fromTime>2001-12-29T00:00:00Z</fromTime>
            <toTime>2001-12-30T23:59:59Z</toTime>
        </TimeRange>
        <Entities>
            <Entity>
                <identifier>100</identifier>
                <Power>
                    <time>2001-12-17T09:30:47Z</time>
                    <value>22.5</value>
                </Power>
            </Entity>
        </Entities>
    </CreateSchedule>
</SubmitForecast>
```

Except for the XML elements bolded below, the response to a Query message is identical to the Submit message.

```
<QueryForecastResponse>
    <ScheduleResponse>
        <Schedule>
            <identifier>1</identifier>
            <!-- 'name' included for informational purposes -->
            <name>STWPF-FCSTMW</name>
        </Schedule>
        <TimeRange>
            <fromTime>2001-12-29T00:00:00Z</fromTime>
            <toTime>2001-12-30T23:59:59Z</toTime>
        </TimeRange>
        <TimeInterval>300</TimeInterval>
        <Entities>
            <Entity>
                <identifier>100</identifier>
                <name>Wind Plant 001</name>
                <Power>
                    <time>2001-12-29T00:00:00Z</time>
                    <value>22.5</value>
                </Power>
                ...
            </Entity>
        </Entities>
    </ScheduleResponse>
</QueryForecastResponse>
```

2.7 Query Response Format

Operations in this document are formatted in two distinct ways. One format is specifically for operations that have both submit and query messages, while the other is for operations that simply have a query message. The main difference between these two formats, aside from the submit/query message format having a submit message, is how the response message is documented in the Data Returned section. The submit/query message format shows the data returned within the response message, including the elements relevant to that message. The query message only format has the data returned by a response message, as well as the full SOAP response message with a sample response message. An example of each Data Returned section format is shown below.

Example: Query Message Only Data Returned Section

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QueryForecastResponse	PowerScheduleResponseType	The outer most element containing all query results
No	No	ScheduleResponse	PowerScheduleDataType	The container element for each schedule returned; results are unbounded
Yes	No	TimeRange	DateRangeType	Contains the time range applied as a filter in the request.
No	No	TimeInterval	long	Describes the amount of time (in seconds) for the time-series data that follows (i.e., resolution)
No	No	name	string	The human readable "name" of the forecast schedule
No	No	time	dateTime: YYYY-MM-DDThh:mm:ss(Z|(+|-)hh:mm)	The "time" element captures the time the value was recorded
No	No	value	decimal	The "value" element contains the value of the measurement.

Full SOAP Message

```xml
<soap:Envelope
      xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <QueryForecastResponse
            xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <ScheduleResponse>
            <Schedule>
               <identifier>?</identifier>
               <name>?</name>
            </Schedule>
            <TimeRange>
               <fromTime>?</fromTime>
               <toTime>?</toTime>
            </TimeRange>
            <TimeInterval>?</TimeInterval>
            <Entities>
               <Entity>
                  <identifier>?</identifier>
                  <name>?</name>
                  <Power>
                     <time>?</time>
                     <value>?</value>
                  </Power>
               </Entity>
            </Entities>
         </ScheduleResponse>
      </QueryForecastResponse>
   </soap:Body>
</soap:Envelope>
```

Sample of Query Submittal Response

```xml
<QueryForecastResponse>
   <ScheduleResponse>
      <Schedule>
         <identifier>9905001</identifier>
         <name>STWPFCST-MW</name>
      </Schedule>
      <TimeRange>
         <fromTime>2012-09-05T01:15:00-07:00</fromTime>
         <toTime>2012-09-05T05:10:01-07:00</toTime>
      </TimeRange>
      <TimeInterval>300</TimeInterval>
      <Entities>
         <!-- 0 to unbounded Entity elements -->
         <Entity>
            <identifier>100</identifier>
            <name>Wind Plant 001</name>
            <!-- 0 to unbounded Power elements -->
            <Power>
               <time>2012-09-05T08:15:00Z</time>
               <value>1</value>
            </Power>
         </Entity>
      </Entities>
   </ScheduleResponse>
</QueryForecastResponse>
```

3. Data Restrictions and Validation

In order for data to submit without error, basic validations must first be met. This chapter describes the validations/restrictions for data and messages that need to be met in order to submit messages to the **e-terra***renewableplan* application.

Basic validations are restrictions on data values submitted, ensuring the data is submitted at the right time according to market rules, and is submitted in the correct format/range. The following sections, "Data Type Validation" highlight these basic validations. This chapter is intended to outline the universal data restriction/validations necessary for submission of a message to the **e-terra***renewableplan* application.

It is recommended that the Data Type Validation sections be printed and used in parallel with constructing any web service operation messages. This will make referencing validations and value restrictions quick and simple.

3.1 Data Type Validation

A basic type of data restriction/validation involves Data Types. A Data Type has restrictions associated with it that are used for submittal messages. A Data Type defines/restricts the range and format of numbers and strings. Data Types are defined and used in the **e-terra***renewableplan* XSD file (see section 1.4), which is linked to both of the **e-terra***renewableplan* WSDL files that provides the Web Service operation messages a participant submits/receives.

An individual Data Type is associated with a specific element or attribute of a web service operation message. Each attribute and element that is in a given message is listed, in this document, in a table located in the "Mandatory and Optional Fields" section of the message. This table not only shows the elements and attributes of an operation, but the data type and format associated with the element or attribute listed as well. In the tables, Data Types are displayed in the following format: Data Type; Format. An example of an element/attribute table that can be found throughout this document is shown below:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments

No	No	time	dateTime;
			YYYY-MM-DDThh:mm:ss(Z|(+|-)hh:mm)
				The forecast date and time.

No	No	identifier	string	Uniquely identifies a Category

This table shows the element or attribute as well as the data type associated with it. The "Format" of the Data Type; Format column of the table is a short description of the type of data that is acceptable for submission.

3.1.1 Data Types (table at bottom)

The following are the most common Data Types, and a description of the "Format" that follows them.

- In a “boolean” data type, either "true" or "false" is entered. This is indicated one of four ways in an element or attribute; true, false, 1, 0. True and the number one are equivalent, while false and the number zero are equivalent.
- In a “dateTime" data type, the format YYYY_MM_DDThh:mm:ss(Z|(+|-)hh:mm); tells that the time must be submitted in an hour:minute:second format with an attached hour:minute adjustment for a time zone preceded with a date in the same format as a Date data type and an intervening "T" character. An example of a time submitted for four o’clock P.M. on July 7, 2010 with a four hour time zone offset is 2010-07-07T16:00:00-04:00.

3.1.1.1 Native XML Data Types

The table below shows the most common native XML data types and a brief description showing the exact strings, values, and/or ranges of data that can be submitted.

Note: The following types are native XML types. All other types are specifically defined for **e-terra***renewableplan*

Data Type	Description
boolean	boolean. values are; true, 1, false, 0
dateTime	The general format is YYYY-MM-DDThh:mm:ss(Z|(+|-)hh:mm); time must use 24-hour format and may not be negative.
decimal	Real number used for telemetry time-series data values. A real number, which can be represented by decimal numerals and (+) positive value is assumed if missing (as defined by the IEEE).
float	32-bit floating-point numbers (as defined by the IEEE)
long	Integer value used primarily for ID's
string	General purpose string (as defined by the IEEE)

3.1.1.2 Common Objects Data Types

The following types are defined in the *CommonObjects Definition section* containing Data Type definitions referenced and extended by the other XML Schema Definition sections. All data types included in the *CommonObjects Definition section* can be considered reusable types that are referenced by at least two data types found in different XSD sections.

The documentation regarding the common object data types is presented in the following sections: first, a table containing a written description for each data type and its children; second, diagrams to help the reader understand the composition and cardinality between data types.

3.1.1.2.1 Data Type Description in Tabular Format

In the following table, the "Data Type" column provides the name of the data type, the "Children: Data Type" column specifies the children elements define contained within the parent, and the "Description" gives a brief description of the data type.

With respect to the "Children: Data Type" column example "fromTime: dateTime" specifies that the element name is "fromTime" and it is uses a "dateTime" data type.

Data Type	**Children: Data Type** **Indicates a required element*	**Description**
BaseRequestType	Category: CategoryIdentityType Schedule: ScheduleIdentityType* TimeRange: DateRangeType Entities: EntitiesIdentityType*	Contains the elements that may be included in most requests, many of which are optional and usage depends on the specific use case.
CategoriesIdentityType	Category: CategoryIdentityType	Container element for a collection of Category elements.
CategoryIdentityType	identifier: string* name: string description: string	Contains information identifying a Category. The ‘identifier’ element is required to uniquely identify a Category and is a string The “name” and “description” elements are meant to be immediately intelligible to a person and should be used for only informational purposes. The “name” element will be present for each query response.
DateRangeType	fromTime: dateTime* toTime: dateTime*	Contains the dateTime elements representing a time range. The “fromTime” element contains the beginning time of the range. The “toTime” element specifies the ending time in the time range. The time range is not constrained and depends largely on the use case. “DateRangeType” values used in queries (i.e., “TimeRange”) retain the original time zone in the response message.

EntityIdentityType	identifier: long assetIdentifier: long name: string description: string isReadOnly: boolean	Contains information identifying an Entity. Either the 'identifier' or the 'assetIdentifier' element is required to uniquely identify an Entity. They can both be provided. The "name" and "description" elements are meant to be immediately intelligible to a person and should be used for only informational purposes. The "name" element will be present for each query response. The "isReadOnly" element should be used by integrating applications to capture their access to a particular Entity. A "false" value is interpreted that they can submit as well as query; whereas "true" is query only.
EntitiesIdentityType	Entity: EntityIdentityType	Container element for a collection of Entity elements.
FaultType	error: TransactionVariance	Web service transactions that fail will return an "error" element containing specific information about the nature of the failure.
PowerEntityIdentityType [extends EntityIdentityType]	Power: TimeValueSeriesType	In addition to the elements contained within the EntityIdentityType, adds a "Power" element that contains time-series data related to power generation.

ScheduleIdentityType	identifier: long* name: string description: string isReadOnly: boolean	Contains information identifying a Schedule. The application represents forecasts and telemetered values as ‘Schedules’. The ‘identifier’ element is required to uniquely identify a Schedule. The “name” and “description” elements are meant to be immediately intelligible to a person and should be used for only informational purposes. The “isReadOnly” element should be used by integrating applications to capture their access to a particular Schedule. A “false” value is interpreted that they can submit as well as query; whereas “true” is query only.
SchedulesIdentityType	Schedule: ScheduleIdentityType	Container element for a collection of Entity elements
ScheduleRequestType	ScheduleRequest: BaseRequestType*	Container element wrapping the request for Schedules.
StatusResponseType	Success: SuccessStatusType	Contains a single “Success” element – which signifies a successful transaction.
SuccessStatusType	transactionId: string*	A “transactionId” uniquely identifies a successful transaction.
SubmitStatusResponse	SubmitStatusResponse: StatusResponseType	Wraps a “Success” element for each successful transaction.
TimeValueSeriesType	time: dateTime value: decimal*	Contains time-series data. The “time” element captures the time the “value” was contains the value of the measurement.
TransactionIdType	transactionId: string*	The “transactionId” element uniquely identifies an individual transaction.

TransactionVariance	message: string* number: int*	Specifies errors related to a failed transaction The error will have an associated unique number which is assigned to the message based on its placement within a collection of messages. If there are 20 “message” elements present, the value of the fifth element within that collection will be 5.

3.1.1.3 Common Operations Data Types

The following data types are defined in the *CommonOperations Definition section.* Elements using these types are referenced in *WindIntegration.xsd*. These data types provide access to data describing the Categories, Schedules and Entities.

The documentation regarding the common operation data types is presented in the following sections: first, a table containing written description for each data type and its children; second, diagrams to help the reader understand the composition and cardinality between data types.

3.1.1.3.1 Data Type Description in Tabular Format

In the following table, the “Data Type” column provides the name of the data type, the “Children: Data Type” column specifies the children elements define contained within the parent, and the “Description” gives a brief description of the data type.

For example “identifier: string” specifies that the element name is “identifier” and it is a “string” data type and it is included in each instance of “CategoryIdentityType”.

Data Type	**Children: Data Type** **Indicates a required element*	**Description**
QueryCategoriesResponseType	Categories: CategoriesIdentityType*	The response data type that contains the collection Category elements. See section 3.1.1.2
QueryEntitiesResponseType	Category: CategoryIdentityType Entities: EntitiesIdentityType*	The response data type that contains the queried for Categories and Entities. See section 3.1.1.2

QueryEntitiesType	Category: CategoryIdentityType	Queries for Entities using a Category identifier as a filter. See section 3.1.1.2
QuerySchedulesResponseType	Schedules: SchedulesIdentityType*	The response data type that contains the queried for Schedules. See section 3.1.1.2

3.2 Handling Times

Within the XML schema, the data type shown in the table below represents date and time values. The lexical representation for this data types is specified by the ISO 8601 standard. The table shows the common notation, though the standard allows for flexibility in formats.

Data Type	Lexical Representation	Example
dateTime	YYYY-MM-DDThh:mm:ss(Z|(+|-)hh:mm);	2011-02-01T11:00:00Z

Note that the dateTime format includes the time zone indicator and the example shows time in Coordinated Universal Time (UTC). XML submissions to the ISO New England **e-terra***renewableplan* are required to specify a time zone indicator to avoid confusion during the daylight savings transition periods. Messages omitting the time zones will fail validation and be rejected.

Adding "-04:00" and “Z” to the end of dateTime representations will specify Eastern Standard Time and UTC, respectively. All samples in documentation will show timestamps that include the time zone. Any dateTime values without the time zone will be rejected and consequently return a Fault response.

All response messages containing date and time values will likewise use the dateTime format outlined above. Importantly, all dateTime values retrieved from a database will be presented with the UTC time zone (conforming exactly to how the value is stored in the database). Query filters (i.e., “TimeRange”) containing dateTime values will retain the original time zone in the response message.

4. Wind Plant Lead Participant Web Service

This section defines the endpoint operations available to Wind Plant Lead Participants. This service differs only from the Forecaster Web Service (section 4.0) in that the Plant Operator Services does not provide Narrative, Telemetry, and Ramp Event operations.

The table below shows a summary of the different messages and indicates if data can be submitted (read & write permissions) or queried (read only permission). The frequency for submitting and querying depends upon the message type; an explanation regarding these time intervals is outside the scope of this document. For more information please consult the *"Wind Integration – Phase 1 Delta Design Note"* documentation.

Message	Brief Description of Usage	**Actions**	
		Submit	Query
Categories	Retrieve a list of Categories. See section 3.1.1.3		X
Schedules	Retrieve an authorized list of Schedules. See section 3.1.1.3		X
Entities	Retrieve a list of authorized Entities; optionally filtered by a Category 'identifier'. See section 3.1.1.3		X
Forecast	Accommodates the requirement to query Wind Plant generation forecasts and submit and query WPFA as a forecast schedule type; includes the Schedule 'identifier', time range, and at least 1 Entity. See section 3.1.1.2.	X	X

4.1 Categories

The Categories message provides the ability to query for `Categories` for which `Entities` are associated. As discussed in the section 3.1.1.3, the response contains information about the available `Categories`; of which 'identifier' values for can be applied as query filters for subsequent operations (i.e., QueryEntities, etc.).

4.1.1 Query Message

4.1.1.1 Purpose of Message

The purpose of this message is to request the collection of all available Categories.

4.1.1.2 Mandatory and Optional Fields

To query for the Categories, the following empty XML element needs to be included in the request:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QueryCategories	QueryCategoriesType	The outermost element which contains an empty sequence.

4.1.1.3 SOAP Format

Full SOAP Message

```
<soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryCategories/>
   </soapenv:Body>
</soapenv:Envelope>
```

Sample of Query Submittal

```
<wint:QueryCategories/>
```

4.1.1.4 Data Returned

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	AuthorizationFault	Fault	An instance of the “FaultType” returned only when a fault occurs. See section 3.1.1.2.1
No	No	QueryCategoriesResponse	QueryCategoriesResponseType	The outermost wrapper identifying the body as the response to the operation invoked.
No	No	Categories	CategoriesIdentityType	See section 3.1.1.3
Yes	No	Category	CategoryIdentityType	See section 3.1.1.3

Full SOAP Format

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <QueryCategoriesResponse
         xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <Categories>
            <!-- 0 or more repetitions -->
            <Category>
               <identifier>?</identifier>
               <name>?</name>
               <description>?</description>
            </Category>
         </Categories>
      </QueryCategoriesResponse>
   </soap:Body>
</soap:Envelope>
```

Sample of Query Response

```
<QueryCategoriesResponse>
   <Categories>
      <!-- 0 or more repetitions -->
      <Category>
         <identifier>WPLANT</identifier>
         <name>WindPlant</name>
         <description>
            A logical grouping of Wind Plants
         </description>
      </Category>
   </Categories>
</QueryCategoriesResponse>
```

4.2 Schedules

The Schedules message provides the ability to query for `Schedules` for which the participant is authorized. As discussed in the section 3.1.1.3, the response contains information about the available `Schedules`; of which ‘identifier’ values will be applied as query filters for subsequent operations (i.e., QueryForecast, etc.).

4.2.1 Query Message

4.2.1.1 Purpose of Message

The purpose of this message is to request the collection of all authorized Schedules.

4.2.1.2 Mandatory and Optional Fields

To query for the Schedules the following empty XML element needs to be included in the request:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QuerySchedules	QuerySchedulesType	The outermost element which contains an empty sequence.

4.2.1.3 SOAP Format

Full SOAP Message

```
<soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QuerySchedules/>
   </soapenv:Body>
</soapenv:Envelope>
```

Sample of Query Submittal

```
<QuerySchedules/>
```

4.2.1.4 Data Returned

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	AuthorizationFault	Fault	An instance of the “FaultType” returned only when a fault occurs. See section 3.1.1.2.1
No	No	QuerySchedulesResponse	QuerySchedulesResponseType	The outermost wrapper identifying the body as the response to the operation invoked.
No	No	Schedules	SchedulesIdentityType	See section 3.1.1.2
Yes	No	Schedule	ScheduleIdentityType	See section 3.1.1.2

Full SOAP Format

```
<soap:Envelope
      xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <QuerySchedulesResponse
            xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <Schedules>
            <!-- 0 or more repetitions -->
            <Schedule>
               <identifier>?</identifier>
               <name>?</name>
               <description>?</description>
               <isReadOnly>?</isReadOnly>
            </Schedule>
         </Schedules>
      </QuerySchedulesResponse>
   </soap:Body>
</soap:Envelope>
```

Sample of Query Response

```
<QuerySchedulesResponse>
   <Schedules>
      <!-- 0 or more repetitions -->
      <!-- Small sample of types represented below -->
      <Schedule>
         <identifier>05010101</identifier>
         <name>STWPFCST-MW</name>
         <description>
            Short Term Wind Power Forecast with 5-minute
            granularity.
         </description>
         <isReadOnly>false</isReadOnly>
      </Schedule>
   </Schedules>
</QuerySchedulesResponse>
```

4.3 Entities

The Entities message provides the ability to query for Entities to which the participant is authorized. As discussed in the section 3.1.1.3, the response contains information about the available Entities; of which ‘identifier’ values will be applied as query filters for subsequent operations (i.e., QueryForecast, etc.).

4.3.1 Query Message

4.3.1.1 Purpose of Message

The purpose of this message is to request the collection of all accessible Entities.

4.3.1.2 Mandatory and Optional Fields

To query for the Entities the following data needs to be included in the request:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QueryEntities	QueryEntitiesType	The outermost element which contains the Category element.
Yes	No	Category	CategoryIdentityType	See section 3.1.1.2; contains the 'identifier' to apply as a filter to the Entity query.

4.3.1.3 SOAP Format

Full SOAP Message

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryEntities>
         <!-- Optional: -->
         <wint:Category>
            <wint:identifier>?</wint:identifier>
         </wint:Category>
      </wint:QueryEntities>
   </soapenv:Body>
</soapenv:Envelope>
```

Sample of Query Submittal

```
<QueryEntities>
   <Category>
      <identifier>WPLANT</identifier>
   </Category>
</QueryEntities>
```

4.3.1.4 Data Returned

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	AuthorizationFault	Fault	An instance of the "FaultType" returned only when a fault occurs. See section 3.1.1.2.1

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QueryEntitiesResponse	QueryEntitiesResponseType	The outermost wrapper identifying the body as the response to the operation invoked. See section 3.1.1.3
No	No	Entities	EntitiesIdentityType	See section 3.1.1.2
Yes	No	Entity	EntityIdentityType	See section 3.1.1.2
Yes	No	Category	CategoryIdentityType	See section 3.1.1.2

Full SOAP Format

```
<soap:Envelope
      xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <QueryEntitiesResponse
            xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <!-- 0 or 1 repetitions -->
         <Category>
            <identifier>WPLANT</identifier>
         </Category>
         <Entities>
            <!-- 0 or more repetitions -->
            <Entity>
               <identifier>100</identifier>
               <assetIdentifier>110</assetIdentifier>
               <name>Wind Plant 01</name>
               <description>Wind Plant 01</description>
               <isReadOnly>false</isReadOnly>
            </Entity>
         </Entities>
      </QueryEntitiesResponse>
   </soap:Body>
</soap:Envelope>
```

Sample of Query Response

```xml
<QueryEntitiesResponse>
   <!-- 0 or 1 repetitions -->
   <Category>
      <identifier>WPLANT</identifier>
   </Category>
   <Entities>
      <!-- 0 or more repetitions -->
      <Entity>
         <identifier>100</identifier>
         <assetIdentifier>110</assetIdentifier>
         <name>Wind Plant 01</name>
         <description>Wind Plant 01</description>
         <isReadOnly>false</isReadOnly>
      </Entity>
   </Entities>
</QueryEntitiesResponse>
```

4.4 Forecast

The Forecast message provides the ability to query and submit forecasts. Participants accessing this endpoint will only have access to the following forecasts:

Query Operations:

- Short Term Wind Plant Forecast
- Medium Term Wind Plant Forecast
- Long Term Wind Plant Forecast

Submit and Query Operations:

- Hourly Wind Plant Future Availability
- Daily Wind Plant Future Availability

Each item in the list above corresponds to a unique Schedule in the **e-terra***renewableplan* system – a listing of which is retrieved using the QuerySchedules operation (see section 4.2). By using the Schedule “identifier” contained within the response from the QuerySchedules operation, forecasts can queried and submitted.

4.4.1 Query Message

4.4.1.1 Purpose of Message

The purpose of this message is to query for submitted forecast data.

4.4.1.2 Mandatory and Optional Fields

To query for the Schedules the following data needs to be included in the request:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	QueryForecast	ScheduleRequestType	The outermost element which contains the ScheduleRequest element.
No	No	ScheduleRequest	BaseRequestType	See section 3.1.1.2
Yes	No	TimeRange	DateRangeType	See section 3.1.1.2 Response is filtered by the provided time range.
Yes	No	Category	CategoryIdentityType	As a convenience, if the query contains a Category and zero Entity elements, the response will contain all Entity elements associated with the Category. See section 3.1.1.2
No	No	Entities	EntitiesIdentityType	See section 3.1.1.2
Yes	No	Entity	EntityIdentityType	If an Entity is included in the query, including the Category is unnecessary. See section 3.1.1.2
No	No	Schedule	ScheduleIdentityType	See section 3.1.1.2

4.4.1.3 SOAP Format

Full SOAP Message

```
<soapenv:Envelope
     xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
     xmlns:wint="wint:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:QueryForecast>
         <!-- 1 or more repetitions: -->
         <wint:ScheduleRequest>
            <wint:Schedule>
               <wint:identifier>?</wint:identifier>
            </wint:Schedule>
            <!-- Optional: -->
            <wint:TimeRange>
               <wint:fromTime>?</wint:fromTime>
               <wint:toTime>?</wint:toTime>
            </wint:TimeRange>
            <wint:Entities>
               <!-- 0 or more repetitions: -->
               <!-- Option 1: list of at least 1 Entity and
                    no Category. Entity can be specified by
                    Identifier, assetIdentifier, or both-->
               <wint:Entity>
                  <wint:identifier>?</wint:identifier>
                  <wint:assetIdentifier>?</wint:assetIdentifier>
               </wint:Entity>
            </wint:Entities>
            <!-- Optional: -->
            <!-- Option 2: all Entities associated with this
                 Category will be included in the response if no
                 entities are provided -->
            <wint:Category>
               <wint:identifier>?</wint:identifier>
            </wint:Category>
         </wint:ScheduleRequest>
      </wint:QueryForecast>
   </soapenv:Body>
</soapenv:Envelope>
```

Sample of Query Submittal

```
<QueryForecast>
   <!-- 1 or more repetitions: -->
   <ScheduleRequest>
      <Schedule>
         <identifier>05010101</identifier>
      </Schedule>
      <!-- Optional: -->
      <TimeRange>
         <fromTime>2012-09-05T15:25:00Z</fromTime>
         <toTime>2012-09-05T19:20:01Z</toTime>
      </TimeRange>
      <Entities>
         <!-- 0 or more repetitions: -->
         <Entity>
            <!-- 1 or both identifiers -->
            <identifier>100</identifier>
            <assetIdentifier>110</assetIdentifier>
         </Entity>
      </Entities>
   </ScheduleRequest>
</QueryForecast>
```

4.4.1.4 Data Returned

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	ForecastFault	Fault	An instance of the "FaultType" returned only when a fault occurs. See section 3.1.1.2.1
No	No	QueryForecastResponse	PowerScheduleResponseType	The outermost wrapper identifying the body as the response to the operation invoked. See section 3.1.1.3
No	No	ScheduleResponse	PowerScheduleDataType	Wrapper for the query response results.
Yes	No	Category	CategoryIdentityType	See section 3.1.1.2
No	No	Schedule	ScheduleIdentityType	See section 3.1.1.2
Yes	No	TimeRange	DateRangeType	See section 3.1.1.2; Response is filtered by the provided time range.

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
Yes	No	TimeInterval	long	For informational purposes, provides the interval, in seconds, applying to all "time" elements in the subsequent time-series data. For example: • 300 = 5 minutes • 3600 = 1 hour • 10080 = 3 days
No	No	Entities	PowerEntitiesIdentityType	Contains Entity elements that have "Power" related time-series data children.
Yes	No	Entity	PowerEntityIdentityType	Extends EntityIdentityType by adding to that definition series of unbounded "Power" elements as children (PowerEntityIdentityType) See (see section 3.1.1.2)
Yes	No	Power	TimeValueSeriesType	Wrapper element for time-series power data. See section 3.1.1.2
No	No	time	dateTime	See section 3.1.1.2
No	No	value	decimal	Value is MW See section 3.1.1.2

Full SOAP Format

```
<soap:Envelope
      xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <QueryForecastResponse
            xmlns="urn:com.alstom.isone.windint.windintegration:1-0">
         <ScheduleResponse>
            <Schedule>
               <identifier>?</identifier>
               <name>?</name>
            </Schedule>
            <TimeRange>
               <fromTime>?</fromTime>
               <toTime>?</toTime>
            </TimeRange>
            <TimeInterval>?</TimeInterval>
            <Entities>
               <Entity>
                  <identifier>?</identifier>
                  <assetIdentifier>?</assetIdentifier>
                  <name>?</name>
                  <Power>
                     <time>?</time>
                     <value>?</value>
                  </Power>
               </Entity>
            </Entities>
         </ScheduleResponse>
      </QueryForecastResponse>
   </soap:Body>
</soap:Envelope>
```

Sample of Query Response

```
<QueryForecastResponse>
   <!-- Example: a short term wind plant forecast -->
   <ScheduleResponse>
      <Schedule>
         <identifier>05010101</identifier>
         <!-- 'name' included for informational purposes -->
         <name>STWPFCST-MW</name>
      </Schedule>
      <TimeRange>
         <fromTime>2012-09-05T15:25:00Z</fromTime>
         <toTime>2012-09-05T19:20:01Z</toTime>
      </TimeRange>
      <!-- Time-series data in 5-minute intervals -->
      <TimeInterval>300</TimeInterval>
      <Entities>
         <Entity>
            <identifier>100</identifier>
            <assetIdentifier>110</assetIdentifier>
            <!-- 'name' included for informational purposes -->
            <name>Wind Plant 01</name>
            <!-- 0 or more repetitions: -->
            <Power>
               <time>2012-09-05T15:25:00Z</time>
               <value>1</value>
            </Power>
         </Entity>
      </Entities>
   </ScheduleResponse>
</QueryForecastResponse>
```

4.4.2 Submit Message

4.4.2.1 Purpose of Message

The purpose of this message is to create a forecast. As mentioned in sections 4.1, 4.2, and 4.3, each forecast submission should contain valid Category (optional), Schedule, and Entity identifiers.

4.4.2.2 Mandatory and Optional Fields

To create a forecast, the following needs to be provided:

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
No	No	SubmitForecast	CreatePowerScheduleType	The outermost wrapper identifying the body as the forecast submission.
No	No	CreateSchedule	PowerScheduleDataType	Wrapper for the forecast submission.
No	No	Schedule	ScheduleIdentityType	See section 3.1.1.2

Opt.	Nil.	Element or Attribute	Data Type; Format	Comments
Yes	No	TimeRange	DateRangeType	See section 3.1.1.2;
No	No	Entities	PowerEntitiesIdentityType	Contains Entity elements that have “Power” related time-series data children.
Yes	No	Entity	PowerEntityIdentityType	Extends EntityIdentityType by adding to that definition “Power” element children (PowerEntityIdentityType) See (see section 3.1.1.2)
Yes	No	Power	TimeValueSeriesType	Container element for time-series power data. See section 3.1.1.2

4.4.2.3 SOAP Format

Full SOAP Format

```
<soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:wint="urn:com.alstom.isone.windint.windintegration:1-0">
   <soapenv:Header/>
   <soapenv:Body>
      <wint:SubmitForecast>
         <!-- 1 or more repetitions: -->
         <wint:CreateSchedule>
            <!-- 1 or more repetitions: -->
            <wint:Schedule>
               <wint:identifier>05180101</wint:identifier>
            </wint:Schedule>
            <!-- Optional: -->
            <wint:TimeRange>
               <wint:fromTime>2012-09-05T15:25:00Z</wint:fromTime>
               <wint:toTime>2012-09-05T19:20:01Z</wint:toTime>
            </wint:TimeRange>
            <wint:Entities>
               <!-- 0 or more repetitions: -->
               <wint:Entity>
                  <!-- 1 or both identifiers -->
                  <identifier>100</identifier>
                  <assetIdentifier>110</assetIdentifier>
                  <!-- 0 or more repetitions: -->
                      <Power>
                         <time>2012-09-05T15:25:00Z</time>
                         <value>1</value>
                      </Power>
               </wint:Entity>
            </wint:Entities>
         </wint:CreateSchedule>
      </wint:SubmitForecast>
   </soapenv:Body>
</soapenv:Envelope>
```

Sample of a Submittal

```
<SubmitForecast>
   <!-- 1 or more repetitions: -->
   <CreateSchedule>
      <!-- 1 or more repetitions: -->
      <Schedule>
         <identifier>05180101</identifier>
      </Schedule>
      <!-- Optional: -->
      <TimeRange>
         <fromTime>2012-09-05T15:25:00Z</fromTime>
         <toTime>2012-09-05T19:20:01Z</toTime>
      </TimeRange>
      <Entities>
         <!-- 0 or more repetitions: -->
         <Entity>
            <!-- 1 or both identifiers -->
            <identifier>100</identifier>
            <assetIdentifier>110</assetIdentifier>
            <!-- 0 or more repetitions: -->
            <Power>
               <time>2012-09-05T15:25:00Z</time>
               <value>1</value>
            </Power>
         </Entity>
      </Entities>
   </CreateSchedule>
</SubmitForecast>
```

Response Message

This message has a standard response as defined in section 2.1.