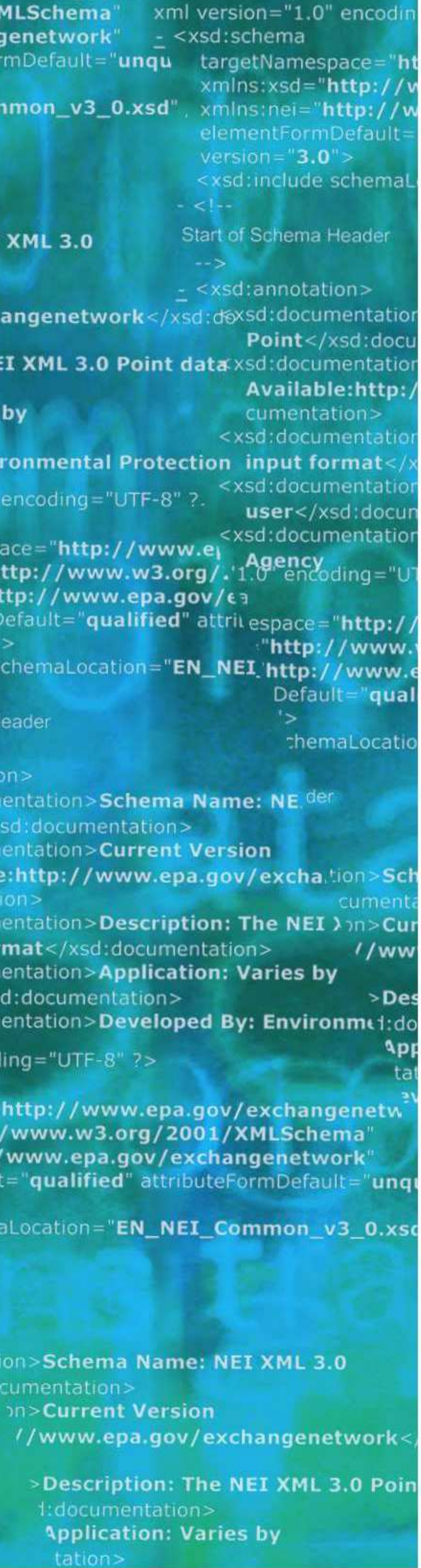

OpenNode2

REST Service Specification (.NET and Java)

Revision Date: 8/12/2013

Prepared By:

WINDSOR SOLUTIONS

4386 SW Macadam Ave, Suite 101
Portland, OR 97239
(503) 675-7833

Environmental Information eXchange Network

Revision History

Date	Author	Changes	Version
6/12/2013	Windsor	Initial Version	1.0
8/12/2013	Windsor	Corrections and clarifications to parameter encoding section	1.1

Table of Contents

Overview

Intended Audience

This document is intended for technical users. It is assumed that the reader is familiar with the fundamental operation of the Exchange Network as well as basic knowledge of SOAP and REST web services technology. This document does not provide in-depth background on the Exchange Network or SOAP/REST.

This document is designed for technical staff that wish to:

- Understand how OpenNode2 implements REST services
- Build a client application that queries and consumes data exposed by OpenNode2
- Build an OpenNode2 plugin that makes available data via the OpenNode2 REST interface

SOAP and REST on the Exchange Network

Web services enable machine-to-machine communication over the Internet. When the Exchange Network specifications were developed in the early 2000's, the dominant web service technology was Simple Object Access Protocol (SOAP). Since that time, Representative State Transfer (REST) services have overtaken SOAP as the most commonly used web service technology.

This is not to say that REST is a superior technology. Both SOAP and REST technologies have advantages and disadvantages. SOAP is and will continue to serve as the backbone of the Exchange Network, however REST services are advantages for certain types of situations. Specifically, REST services are best suited for synchronous queries. **Simply put, REST makes it easy to query data from OpenNode2 using only a URL.** Without REST, a data requestor would need to build a SOAP request; a task that usually requires specialized toolkits or frameworks to construct the query and consume the SOAP response back from OpenNode2.

It is technically possible for RESTful services to perform other types of Exchange Network operations other than synchronous queries, but at this time the OpenNode2 REST capabilities are limited only to performing queries.

Additional Information

REST support in OpenNode2 is based on the approved Exchange Network REST Guidance documentation available online at:

http://www.exchangenetwork.net/rest-guidance/

OpenNode2 REST Service URL syntax is modeled after the CDX Rest Proxy URL structure described in Section 6 of the REST guidance document referenced above.

OpenNode2 REST Specification

The OpenNode2 REST interface provides an alternate mechanism to performing a SOAP query. OpenNode2 plugins do not need to be custom tailored to support REST queries. If an OpenNode2 plugin implements a Query operation, OpenNode2 will make this available via both the traditional SOAP endpoints as well as the REST endpoint.

REST services are supported on the following OpenNode2 platforms:

- .NET OpenNode2 v2.6 and higher
- Java OpenNode2 v2.08 and higher

Request URL Format

Generally speaking, OpenNode2 REST Service URL syntax is modeled after the CDX Rest Proxy URL structure described in Section 6 of the official Exchange Network REST guidance document referenced in the previous section. The sections below provide detailed information on using constructing a RESTful query to OpenNode2.

RESTful queries are available for all OpenNode2 plugin query services. The downloadable OpenNode2 plugin installation guides will be updated to provide examples of REST URL query syntax for query requests supported by each flow. For flow-specific examples, please view the applicable OpenNode2 Plugin Installation Guide.

The OpenNode2 REST URL syntax is as follows:

> *NodeURL*/*RestEndpoint*/**Query?Dataflow=***dataflow***&Request=***request*[**&Username=***username* **&Password=***password*][**&token=***token*][**&ZipResults=***TrueOrFalse*][**&Params=***params*]

Optional parameters are shown in [brackets]. Static portions of the URL are shown in **bold** face.

Parameter	Description	Example Value
NodeURL	The base URL of the OpenNode2 instance	https://www.myagency.gov/node
RestEndpoint	The URL path to the REST endpoint. While the OpenNode2 installation guide recommends an endpoint application name of "REST", agencies may choose to use a different application name.	REST
dataflow	The Exchange Network data flow to query. The name must match exactly with the standard flow identifier prescribed in the FCD for the exchange. OpenNode2 administrators may choose	FacID_v3.0
request	The name of the query to invoke. The value provided should match the query service name defined in the FCD for the given data exchange (dataflow).	GetFacility_v3.0
username	The NAAS username to use to authenticate to OpenNode2.	me@myagency.gov

password	The NAAS password to use to authenticate to OpenNode2.	MyP@ssword1!
token	In lieu of explicitly supplying a username and password, a valid NAAS-issued token can be provided. This allows a client to authenticate to NAAS directly instead of by proxy via the OpenNode2 REST query.	FTOVMp9-D_L7LadLZmT0…
TrueOrFalse	Set to True if zipped response is to be returned from OpenNode2. The default is false if this parameter is not supplied in which case the response will be XML. Note that	True
Params	The query parameters (filters) to apply. The parameters supported are specific to the data flow and service. Please see the section below for the syntax requirements for defining parameters.	Facility Name|ACME%; Zip Code|48123

Example

Two example REST queries are shown below. Please note that valid NAAS credentials must be substituted.

https://www.windsorsolutions.biz/OpenNode2_dotNet/RestEndpoint/query?Dataflow=FacID_v3.0&Request=GetFacility_v3.0&Username=cdx&Password=test&ZipResults=false&Params=Change%20Date|1/1/2005;Facility%20Name|F%25;SIC%20Code|3089|6531|3643

Query Parameters

Parameters are specific to the query being invoked. The Exchange Network FCD for the given dataflow should be consulted to learn the specific parameters supported by a given query. Parameters are supplied in the *Params* URL parameter. Some queries may not require any parameters, some parameters may be required, and some parameters may repeat and some may only allow one value to be passed. Again, the FCD should be consulted to determine the allowable parameter combinations.

Parameters must be passed using the following rules:

- Parameters are passed as name/value pairs separated by a pipe symbol (|)
- Each name/value pair is separated by a semicolon (;)
- To pass multiple values for the same parameter, add the parameter twice using the same name but with the different values in each parameter. This is an implicit OR condition. For example, to query for records in two zip codes the syntax would be *Zip Code|48111;Zip Code|48112*.

Parameter Encoding Considerations

URLs can only be sent over the Internet using a subset of the ASCII character-set[1]. Since typed URLs can contain these "unsafe" and/or reserved characters (spaces, colons, brackets, etc.), the URL has to be

[1] http://www.blooberry.com/indexdot/html/topics/urlencoding.htm

encoded into a valid format. URL encoding replaces unsafe ASCII characters with a "%" followed by two hexadecimal digits.

For example the text:	Zip Code|48111;Zip Code|48112
is encoded as:	Zip%20Code%7C48111%3BZip%20Code%7C48112

If the OpenNode2 REST service is invoked by a person using a web browser, the browser will automatically encode the URL. The encoding may or may not be displayed in the browser's address bar depending on the browser. However, if calling the OpenNode2 REST service via JavaScript/AJAX, the encoding must be supplied in the request. Online tools such as Free Formatter (http://www.freeformatter.com/url-encoder.html) can be used to encode URL strings.

Wildcards in search criteria must also be considered. In the FacID query example above, the criteria specifies that the service should only return facilities that start with the letter "F". This requires the user to add the percent symbol (F%) to the facility name search string to indicate a wildcard search. Since the percent symbol is considered an unsafe character, it must also be encoded with the hex value %25 resulting in "F%25" in the example.

User Authentication

OpenNode2 currently requires NAAS authentication for all requests. For this reason, the request must contain either a username/password or a token. Future versions of OpenNode2 may support non-authenticated (public) services.

Note that security on exchanges is set by the node administrator. Contact the node administrator if you need access to a given service. If you do not have permission, an E_AccessDenied error will be returned.

The .NET OpenNode2 in particular will prompt the user for credentials using HTTP basic authentication[2] if none are supplied in the request. Java OpenNode2 does not contain this feature.

Specifying Result Format

Results can be returned in XML or ZIP format. ZIP is returned when ZipResult=True is present in the request URL, otherwise XML is returned.

The HTTP header Accept parameter is not used by OpenNode2 to determine the result format returned to the client.

Response Format

The response returned from OpenNode2 will either be an XML document conforming to the XML schema defined for the exchange or an error. Errors returned in XML and are formatted as follows:

```
<Error>
  <Message>Exception: java.lang.RuntimeException,Message: Error while validating:
Error while validating token: Security Token Error
  </Message>
</Error>
```

[2] http://en.wikipedia.org/wiki/Basic_access_authentication