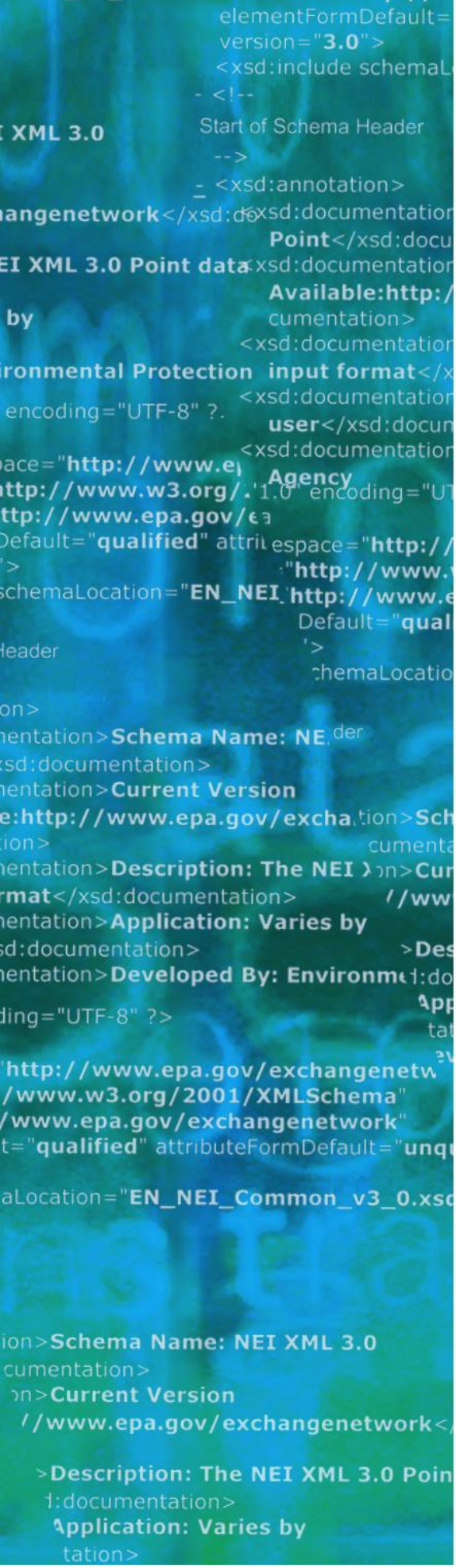


Water Quality Exchange

Flow Configuration Document

Version 2.1c

October 15, 2015

Developed for:

United States Environmental Protection Agency
Office of Environmental Information
1200 Pennsylvania Avenue, NW
Washington, DC 20460

Developed by:

Gold Systems, Inc.
2121 S. McClelland St., #204
Salt Lake City, Utah 84106

Environmental Information exchange Network

Table of Contents

1.0 Introduction

1.1 Background

The U.S. Environmental Protection Agency (EPA) Office of Information Collection (OIC), Office of Wetlands, Oceans and Watersheds (OWOW), and Office of Water (OW) are committed to implementing Central Data Exchange (CDX) services and establishing the EPA infrastructure to support an ambient water quality monitoring data exchange. The Water Quality Exchange project is the product of a collaborative effort between OIC, OW, and the Environmental Council of States (ECOS). The project was identified during the development of the Environmental Sampling Analysis and Results (ESAR) data standard for water monitoring.

The project goal is to provide EPA state partners with a means of exchanging water quality monitoring data via CDX, utilizing the ESAR data standard as much as possible. The Office of Water, in partnership with the states, establishes water quality monitoring data exchange elements, business rules for exchanging these elements, and valid domain lists for elements not covered by an existing or proposed standard.

The WQX System includes the following types of data:

- The physical conditions in the environment at the time of a site visit.
- The chemical and bacteriological make-up of the water sampled.
- Chemical analyses of fish tissue collected.
- Biological Taxon Abundance data, including population census, frequency class, group summaries, and individual results
- Toxicity data.
- Habitat Assesment scores and their related metric scores.
- Biological Index scores and their related metric scores.

The WQX data flow utilizes two mechanisms for exchanging water quality monitoring information:

- Web services-based solution for automated submission utilizing Exchange Network standards.
- Web-based solution for manual submissions (included with an XML Generation Tool to be developed at a later date).

1.2 Document Purpose

This Flow Configuration Document (FCD) is intended to define the supported data services, the approaches and processes that are used to exchange information over the Exchange Network via web services technology. In addition, the FCD serves as a guide for trading partners in understanding the details and challenges associated with a data flow.

The purpose of this FCD is to describe the operation of the Water Quality Monitoring Network Exchange using XML-based data submissions through Node-to-Node or Client-to-Node transfers. This document does not consider use of the alternative ESAR formats or transfer mechanisms. The focus of the document is the core WQX data flow between the state, local, and tribal agencies and EPA.

The Office of Water is separately committed to providing outbound services from the STORET Data Warehouse, which combines WQX data and STORET data. These data services are outside the scope of this document.

1.3 How to use this FCD

This FCD provides guidance to implement an XML/web-service based model for data submission. For the WQX project, there are two types of submissions, Update-Insert and Delete. This FCD expands upon the usage of the WQX Schema and introduces the implementation of the Document Header.

This document includes the following main sections:

<u>Implementing the Header Document</u>

This section describes how the WQX Network Exchange makes use of the Exchange Network Header Document to describe the payload content of a Network message. This submission structure is used for both Node-to-Node and Client-to-Node submissions as well as the web submission (to be implemented at a later phase).

<u>Data Processing</u>

This section describes the data processing rules in effect in the WQX System.

<u>Submission Processing and Feedback</u>

This section describes the processing steps and status changes that occur as your submission flows through CDX and the WQX System.

Web Service Methods at CDX

This section describes the Web Service Methods at CDX used by the WQX Data Flow.

2.0 Component Alignment and Change History

2.1 Flow Component Version History

Component	Version	Date	Changed By	Description of Change
Flow Configuration Document (FCD)	1.0	11/15/2006	Ryan Jorgensen	Original Version
Schema	1.0	1/19/2007	Doug Timms	Original Version
Data Exchange Template (DET)	1.0	1/19/2007	Doug Timms	Original Version
GetActivityByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetActivityGroupByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetMonitoringLocationByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetProjectByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetResultByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetDomainValueByElementName data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
GetTransactionHistoryByParameters data service	1.0	2/1/2007	Ryan Jorgensen	Original Version
FCD	2.0	06/20/2008	Ryan Jorgensen	Major revision for WQX 2.0
Schema	2.0	06/04/2008	Doug Timms	Major revision for WQX 2.0
DET	2.0	06/04/2008	Doug Timms	Major revision for WQX 2.0
GetActivityByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Added support for new WQX 2.0 elements
GetActivityGroupByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Changed schema reference to wqx/2. No other changes.
GetBiologicalHabitatIndexByParameters data service	2.0	07/10/2008	Ryan Jorgensen	New data service for WQX 2.0
GetMonitoringLocationByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Added support for new WQX 2.0 elements
GetProjectByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Added support for new WQX 2.0 elements

GetResultByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Added support for new WQX 2.0 elements
GetDomainValueByElementName data service	2.0	07/10/2008	Ryan Jorgensen	Changed schema reference to wqx/2 and added new 2.0 domain value lists.
GetTransactionHistoryByParameters data service	2.0	07/10/2008	Ryan Jorgensen	Changed schema reference to wqx/2. No other changes.
FCD	2.1	11/11/2009	Ryan Jorgensen	Minor revision for WQX 2.1
Schema	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
DET	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetActivityByParameters data service	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetActivityGroupByParameters data service	2.1	11/11/2009	Ryan Jorgensen	No Change
GetBiologicalHabitatIndexByParameters data service	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetMonitoringLocationByParameters data service	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetProjectByParameters data service	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetResultByParameters data service	2.1	11/11/2009	Ryan Jorgensen	Increased max length on some elements
GetDomainValueByElementName data service	2.1	11/11/2009	Ryan Jorgensen	No Change
GetTransactionHistoryByParameters data service	2.1	11/11/2009	Ryan Jorgensen	No Change
FCD	2.1b	06/12/2012	Ryan Jorgensen	Changed max submission file size to 350MB (unzipped). Changed rule #15. Corrected parameter spelling organizationIdentifer to organizationIdentifier

FCD	2.1c	10/15/2015	Ryan Jorgensen	Revised section 4 to clarify WQX behavior with respect to updating existing data

2.2 Flow Component Versions Currently Supported

Component	Version(s) Supported	Explanation (optional)
FCD	2.1	
Schema	1.0, 2.0, and 2.1	1.0, 2.0, and 2.1 supported for data submissions. Most data services (i.e. query/solicit) only support WQX 2.0 and 2.1
DET	1.0, 2.0, and 2.1	
GetActivityByParameters data service	2.0 and 2.1	
GetActivityGroupByParameters data service	2.0 and 2.1	
GetBiologicalHabitatIndexByParameters data service	2.0 and 2.1	
GetMonitoringLocationByParameters data service	2.0 and 2.1	
GetProjectByParameters data service	2.0 and 2.1	
GetResultByParameters data service	2.0 and 2.1	
GetDomainValueByElementName data service	1.0, 2.0, and 2.1	
GetTransactionHistoryByParameters data service	1.0, 2.0, and 2.1	

2.3 Business Rule Change History

Business Rule Change	Date of Change	Explanation (optional)
When SampleTissueTaxonomicName or SampleTissueAnatomyName is reported, both must be reported.	WQX 2.0 (July 2008)	Rule dropped
When ResultDetectionConditionText is 'Not Detected', DetectionQuantitationLimitTypeName and DetectionQuantitationLimitMeasure must be reported	WQX 2.0 (July 2008)	Expanded to include 'Present Above Quantification Limit' and 'Present Below Quantification Limit'
Element name for "SampleTissueTaxonomicName" was	WQX 2.0 (July 2008)	SampleTissueTaxonomicName and SampleTissueAnatomyName

changed to "SubjectTaxonomicName" in WQX 2.0.		moved from being part of an Activity in WQX 1.0 to part of a Result in WQX 2.0. Taxonomic Name is no longer only used for tissue samples so the name was changed to SubjectTaxonomicName
Additional Business Rules have been created that relate to new elements in WQX 2.0.	WQX 2.0 (July 2008)	See section 4.1.1 for a complete list of rules.
Rule #15 changed to "CharacteristicName and ResultStatusIdentifier must be reported"	WQX 2.1 (June 2012)	

3.0 Implementing the Header Document

3.1 Overview

The Exchange Network Header Document provides additional information about the contents of a message payload. It is developed to further automate the data exchange process so that data can be more readily identified during transport and managed at its processing destination. The Header Document can describe what a data payload contains, who submitted it, when it was submitted, as well as instructions on processing the payload contents, such as whether the contents are inserts and updates or deletions.

Essentially, the Header Document is an XML wrapper, placed around an XML payload before transmission to CDX. A Header Document toolkit is available in the Tool Box section of the Exchange Network Web site (http://www.exchangenetwork.net), containing additional background about the Header Document, as well as Java and .NET implementation tools.

3.2 Header Document Structure

Any network exchange for WQX must use the Header Document structure in order to meet EPA CDX and WQX processing requirements and prior Exchange Network agreements.

The Header Document begins with a "Document" tag and contains two sections: "Header" and "Payload". The Header section contains information about the submission. The Payload contains the WQX data. Although some data flows allow for multiple payloads in one document, the WQX data flow allows only a single payload. The Payload must conform to one of the two WQX standard schemas for data submissions.

The following table describes the Header Document elements and attributes and how they are utilized for the purpose of a WQX submission.

Document				
Attribute	**Description**	**Example Value**	**Required**	**Notes**
Id	Allows a submitter to provide a unique ID for each submission.	"ABC12345"	Yes	It is not mandatory that this value be unique.
Header Section				
Element	**Description**	**Example Value**	**Required**	**Notes**
Author	First and Last Name of individual generating the XML document	Joe Smith	Yes	Reference only

Organization	Name of company, environmental agency or individual that created the XML document	State X Department of Environmental Quality	Yes	This identifies the author's organization, which may be different from the organization who owns the data (identified in the WQX payload)
Title	Identifies the target data flow for a submission	"WQX"	Yes	
Creation Time	Date/Time when the document was generated	2003-01-01T12:12:12 (where date is a valid XML date format string)	Yes	
Comment	Free text description of the message contents.		No	
Data Service	Unused	N/A	No	Element is not used. If a value is included it will be ignored.
Contact Info	Name, mailing address, city, state, zip, telephone number, and email address of person who may be contacted with questions concerning the submission.	Joe Smith 123 Main St. Portland, OR 97226 503 123 4567 Joe@deq.statex.gov	Yes	
Notification	This element is used only by the CDX Node (internally).	N/A	No	Element is not used. If a value is included it will be ignored.
Sensitivity	Unused	N/A	No	Element is not used. If a value is included it will be ignored.
Property	Unused	N/A	No	Element is not used. If a value is included it will be ignored.
Payload				
Attribute	**Description**	**Example Value**	**Required**	**Notes**
Operation	This describes the operation to be performed on the payload. Multiple values are not allowed	Must be either “Update-Insert” or “Delete”	Yes	
WQX				
XML Namespace (xmlns)	WQX approved namespace for submission files	Must be either: "http://www.exchangenetwork.net/schema/wqx/1"	Yes	To take advantage of the new elements in

		or "http://www.exchangenetwork.net/schema/wqx/2"		the WQX 2.0 Schema you must reference the wqx/2 namespace.

The following excerpt from a WQX XML Submission File demonstrates the use of the header document:

```
<?xml version="1.0" encoding="UTF-8"?>
<Document Id="123456789"
          xmlns="http://www.exchangenetwork.net/schema/v1.0/ExchangeNetworkDocument.xsd"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Header>
    <Author>Ryan Jorgensen</Author>
    <Organization>Gold Systems, Inc.</Organization>
    <Title>WQX</Title>
    <CreationTime>2008-04-05T09:30:47-05:00</CreationTime>
    <Comment>This is a test file</Comment>
    <ContactInfo>2121 S. McClelland St., #204, SLC, UT  84106, 801-456-6105, ryanj@goldsystems.com</ContactInfo>
  </Header>
  <Payload Operation="Update-Insert">
    <WQX xmlns=http://www.exchangenetwork.net/schema/wqx/2
         xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
         xsi:schemaLocation="http://www.exchangenetwork.net/schema/wqx/2
             http://www.exchangenetwork.net/schema/wqx/2/index.xsd">
      <Organization>
        ...
      </Organization>
    </WQX>
  </Payload>
</Document>
```

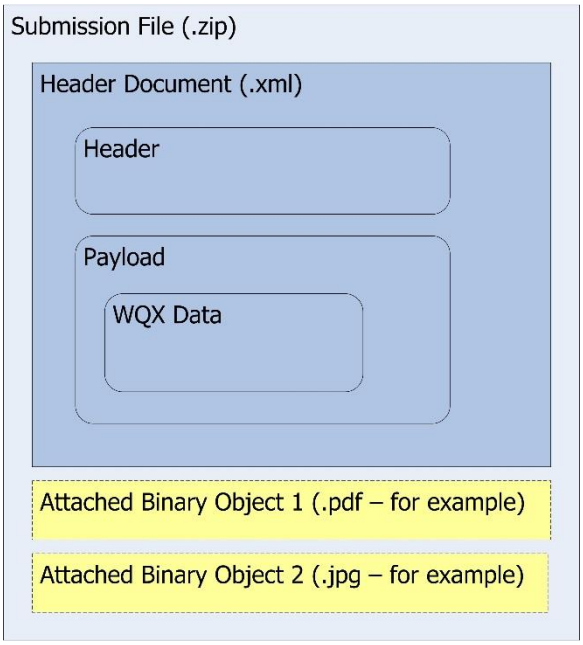
3.3 Submission File Structure

In addition to the XML Header Document (containing the Header and Payload), a submission can also include binary objects (such as images or documents). These binary objects are referenced (by unique name) in the WQX Document, but are provided as individual, external documents. In order to group all related documents for a submission, each WQX submission should be provided to CDX in a single compressed file in '.zip' format. This also improves transmission efficiency due to the significantly smaller size of compressed files. Even if a submission does not contain any external binary objects, it is still strongly recommended that it be provided in '.zip' format (although single '.xml' documents will be allowed).

Additionally, the Header Document within the submission is the only file allowed to have an '.xml' extension. This allows CDX to differentiate between the Header Document and any other binary object files included in the submission.

It is also recommended that submission files be keep files under 350 MB (before they are zipped) or about 350,000 Results.

The following diagram describes the structure of a Submission Zip File for WQX:



3.4 Payload

The Payload Section of the Header Document contains an attribute named “Operation”. This is used to denote the type of processing for a submission. There are two acceptable values: “Update-Insert” or “Delete”. Use of these operators triggers the processing outlined in the *Data Processing* section of this document. For example, a Payload Operation of “Update-Insert” informs the back-end application that the submission contains new data (to be added to WQX) or existing data (to be updated in WQX). An Operation of ”Delete” denotes that the submission contains identifiers for data to be deleted from WQX.

The Payload section of the document contains the WQX data, which must conform to one of the two standard WQX XML Schema Definitions (XSD) for data submissions:

1. If the “Operation” attribute is “Update-Insert” then the Payload must conform to the Update-Insert Schema (“WQX_WQX_v2.1.xsd”).
    a. For backward compatibility WQX will continue to support submissions that comply with the WQX 1.0 schema (“WQX_WQX_v1.0.xsd”) and WQX 2.0 schema (“WQX_WQX_v2.0.xsd”).
2. If the “Operation” attribute is “Delete” then the Payload must conform to the Delete Schema (“WQX_WQX_Delete_v2.1.xsd”).
    a. For backward compatibility WQX will continue to support submissions that comply with the WQX 1.0 schema (“WQX_WQX_Delete_v1.0.xsd”) and WQX 2.0 schema (“WQX_WQX_Delete_v2.0.xsd”).

4.0 Data Processing

Each data submission to WQX includes an "OrganizationIdentifier" element. This is a unique identifier for the organization whose data is to be modified (by the data submission).

Before processing a data submission, the WQX system will confirm that the OrganizationIdentifier is valid and that the user (identified by a NAAS ID) has been authorized to modify data for this organization.

As long as this is true, the WQX System will proceed with processing the data submission. Sections 4.1 and 4.2 provide an overview of the structure and processing related to the two types of payloads for WQX: “Update-Insert” and “Delete”, as well as provide a list of the data rules enforced by the WQX System (which are not described by the respective XSDs).

4.1 Update-Insert

The “Update-Insert” Schema (XSD) is modeled in Exhibit 1. *Note: it may be helpful to print the schema/model (in Exhibit 1) while reading this section.* As the name implies, this schema defines the structure of a submission file which can be used to "Update" or "Insert" (i.e. add) data to the WQX System.

The elements in this schema are grouped into 8 components (represented by the 8 connected boxes in the model). The list of components and their general hierarchy is provided here (*spaces have been added to component names for readability*).

- Organization
    - **Project**
    - **Monitoring Location**
    - **Biological/Habitat Index**
    - **Activity Group**
    - **Activity**
        - Activity Metric
        - Result

The schema defines one root component for the entire hierarchy: Organization. Each submission file must contain one, and only one, Organization (uniquely identified with an OrganizationIdentifier).

Under the Organization, five primary components exist (bolded in the list above and designated with a heavier border on the boxes in the model in Exhibit 1): Project, Monitoring Location, Activity, Biological/Habitat Index and Activity Group. Two secondary components also exist (Activity Metric, and Result), which fall below an Activity.

A valid WQX submission file should include one Organization and one or more of the primary components. Consider the following examples of the types of data that might be included in a valid submission:

- 1 Organization and 3 Projects
- 1 Organization, 5 Projects, and 37 Monitoring Locations
- 1 Organization, 250 Biological/Habitat Indexes, 1320 Activites, and 5400 Activity Metrics

Each primary component in the schema includes an identifier (e.g. ProjectIdentifier, MonitoringLocationIdentifier, ActivityIdentifier, etc.). Each must be unique within the parent Organization. For example: no two Monitoring Locations can have the same identifier and no two Activities can have the same Identifier (within a single submission to WQX). Likewise, it's important to understand the implications of using an identifier in one submission that has been used in a previous submission.

When an identifier (for one of the five primary components) is encountered in a submission, the database will be queried to determine if a component with that identifier can be found (indicating that that identifier has been used in a previous submission). If the identifier is found, then an “Update” is performed, overwriting all of the component details in the database with the component details in the current data submission. If the identifier is not found, an “Insert” is performed, creating an entirely new component record in the database.

Secondary components (i.e. Activity Metric or Result) do not have unique identifiers, and therefore cannot be updated individually. All secondary components in the schema must be updated along with their respective parent. For example: in order to update a Result that has been previously submitted to the WQX System, you must resubmit the parent Activity and all Results that fall below it. To further clarify, suppose you submit an Activity with three Results to WQX. Then you wish to add one new Result to that original Activity. You must submit a file with the one original Activity, the three original Results, plus the one new Result (making four total Results) to WQX.

When a primary component (in the database) is updated (with a component from a new submission), it's important to understand that it is a comprehensive replacement of all the component details (or elements) and any secondary component details that fall below it. There is no way to update just part of the component details. Likewise, there is no way to add additional information to an existing component (just by submitting the new information). You must completely replace all of the details. For example:

- Although a Monitoring Location can have multiple "AlternateMonitoringLocationIdentity" values, you cannot start with a Monitoring Location with one AlternateMonitoringLocationIdentity value, then submit an update to that Monitoring Location with one new AlternateMonitoringLocationIdentity value, and then be left with two AlternateMonitoringLocationIdentity values for that Monitoring Location. Instead, your update to the original Monitoring Location must include two AlternateMonitoringLocationIdentity values: the original one and the new one.
- Likewise, an Activity can have mulitiple ProjectIdentifiers, but you cannot add an additional ProjectIdentifier to an existing Activity. You must resubmit the Activity (to WQX) with two ProjectIdentifies: the original one and the new one.

As the WQX System processes a submission, it saves each primary component once it has passed all validation rules. If there are any errors encountered with the primary component (or any related secondary component), then the primary component all related secondary components will not be saved. For example: if an Activity with five Results is being processed, and one Result (among the five) has a validation error, then the Activity and the five Results will be discarded and will not be saved.

If any validation error was found while loading the data submission into the WQX System, the status for that submission will be set to “Failed” at CDX. This does not mean that there were zero components that were saved. It only means that there was at least one component that was not saved because of a validation error. The Processing Report will include an itemized list of any primary components that failed to be saved. See Appendix A for an example Processing Report.

If a submission fails to process successfully, there are two approaches to correcting it.

1) Correct the validation errors in the original submission and then resubmit the entire submission. In this case, components that were valid in the first submission will be updated by the second submission (with the exact same values as before). Components that failed in the first submission (which were never saved) will now be "Inserted" (i.e. added) by the second submission.
2) Or you can just resubmit the components that failed in the original submission (since that's all that's needed).

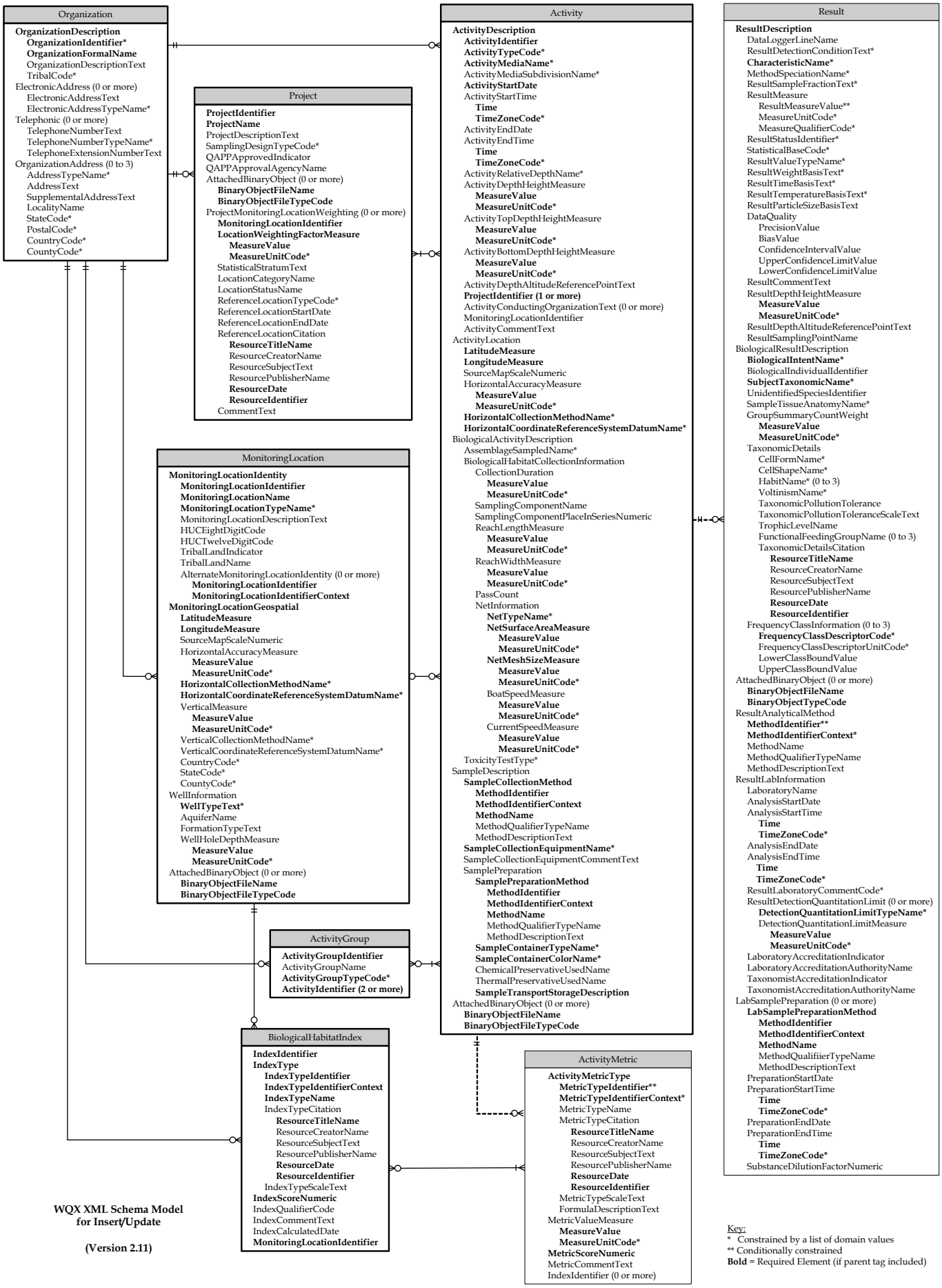


Exhibit 1 – Update-Insert Schema

4.1.1 *Data Rules*

The following is a summary of the data validation rules enforced on an "Insert-Update" submission to WQX. These are in addition to the rules enforced by the WQX XSD.

1. When ElectronicAddressText or ElectronicAddressTypeName is reported, both must be reported.
2. When TelephoneNumberText or TelephoneNumberTypeName is reported, both must be reported.
3. When AddressText or AddressTypeName is reported, both must be reported.
4. When HorizonitalCollectionMethodName is “Interpolation-Map”, SourceMapScaleNumeric must be reported.
5. When VerticalMeasure's MeasureValue is reported, the following also must be reported:
   VerticalMeasure's MeasureUnitCode,
   VerticalCollectionMethodName,
   VerticalCoordinateReferenceSystemDatumName.
6. Either ProjectDescriptionText or Project's AttachedBinaryObject must be reported.
7. Activity Depth/Height can be reported in only one of the following two ways (but not both):
   a. Specific depth using ActivityDepthHeightMeasure's MeasureValue.
   b. Depth Range using ActivityTopDepthHeightMeasure's MeasureValue and ActivityBottomDepthHeightMeasure's MeasureValue.
      i. This method must be used when ActivityTypeCode is “Sample-Integrated Vertical Profile”.
8. When ActivityTypeCode contains the word 'Logger', DataLoggerLineName must be reported.
9. When ActivityTypeCode contains the word 'Sample', SampleCollectionMethod must be reported.
10. When ActivityMediaName is "Tissue" then BiologicalIntentName must also be "Tissue" (and visa-versa)
11. When ActivityMediaName (or BiologicalIntentName) is "Tissue", then SampleTissueAnatomyName must be reported.
12. When ActivityMediaName is "Biological" then AssemblageSampledName must be reported
13. When ResultDetectionConditionText is 'Not Detected', 'Present Above Quantification Limit' or 'Present Below Quantification Limit', then DetectionQuantitationLimitTypeName and DetectionQuantitationLimitMeasure must be reported.
14. Either ResultMeasure's ResultMeasureValue or ResultDetectionConditionText must be reported, but not both.
15. CharacteristicName and ResultStatusIdentifier must be reported.
16. When DetectionQuantitationLimit's MeasureValue is reported, DetectionQuantitationLimit's MeasureUnitCode must be reported.
17. ActivityDescription’s MonitoringLocationIdentifier may be required depending on the value provided for ActivityTypeCode. See the domain value list for ActivityTypeCode for more information.
18. ResultAnalyticalMethod may be required depending on the value provided for ActivityTypeCode. See the domain value list for ActivityTypeCode for more information.
    a. However, ResultAnalyticalMethod is never required if BiologicalIntentName is "Individual", "Population Census", "Frequency Class", or "Group Summary"
19. ResultSampleFractionText may be required depending on the value provided for CharacteristicName. See the domain value list for CharacteristicName for more information.
20. ResultAnalyticalMethod’s MethodIdentifierContext must either match a value from the AnalyticalMethodContext domain list or it must be the same as the value for the OrganizationIdentifier provided in the submission file.
    a. If the MethodIdentifierContext matches a value from the domain list, then the MethodIdentifier must also match a value from the AnalyticalMethod domain list (for that Context). Furthermore, MethodName, MethodQualifierTypeName, and MethodDescriptionText are not required and will be ignored (since only the Identifier and IdentifierContext are needed to uniquely identify the Analytical Method).
    b. If the MethodIdentifierContext matches your OrganizationIdentifier (indicating your own method), then MethodIdentifier and MethodName are both required, but do not need to match a value from the domain list (since they are your own). Additionally, MethodQualifierTypeName and MethodDescriptionText can be provided, but are optional, to further describe the Analytical Method used.

21. ProjectIdentifier, MonitoringLocationIdentifier, ActivityIdentifier, IndexIdentifier and ActivityGroupIdentifier must be unique within an Organization. The value for each of these identifiers may occur only once in a submission file.
    a. Unique identifiers are treated as case-insensitive by WQX. For example, the following three identifiers would be treated as identical: “Mx571”, “mx571”, “MX571”.
22. If Sample Preparation block is reported, then either ChemicalPreservativeUsedName or ThermalPreservativeUsedName must be reported.
23. ResultMeasure's ResultMeasureValue may be constrained to a list of domain values depending on the value provided for CharacteristicName. See the domain value list for CharacteristicName for more information.
24. If a numeric value is reported for ResultMeasureValue, then ResultMeasure's MeasureUnitCode and ResultValueTypeName are required.
    a. The exception to this is when the ResultMeasureValue is a Characteristic Pick List Value. These do not have units.
25. If a CountyCode is reported then a StateCode must also be reported.
26. If NetTypeName = "Net/Horizontal Tow" then BoatSpeedMeasure is required.
27. If NetTypeName is reported then the SampleCollectionEquipmentName must be one that relates to that type of equipment.
28. ActivityMetric's MetricTypeIdentifierContext must either match a value from the MetricTypeContext domain list or it must be the same as the value for the OrganizationIdentifier provided in the submission file.
    a. If the MetricTypeIdentifierContext matches a value from the domain list, then the MetricTypeIdentifier must also match a value from the MetricType domain list (for that Context). Furthermore, MetricTypeName, MetricTypeCitation, MetricTypeScaleText, and FormulaDescriptionText are not required and will be ignored (since only the Identifier and IdentifierContext are needed to uniquely identify the MetricType).
    b. If the MetricTypeIdentifierContext matches your OrganizationIdentifier (indicating your own metric), then MetricTypeIdentifier and MetricTypeName are both required, but do not need to match a value from the domain list (since they are your own). Additionally, MetricTypeCitation, MetricTypeScaleText, and FormulaDescriptionText can be provided, but are optional, to further describe the Metric Type used.
29. If BiologicalIntentName is "Group Summary" then GroupSummaryCountWeight must be reported.
30. If BiologicalIntentName is "Frequency Class" then Result's CharacteristicName must be "Count"
31. If BiologicalIntentName is "Population Census" then Result's CharacteristicName must be "Count" or "Total Sample Weight"
32. FrequencyClassDescriptorUnitCode may be required depending on the value provided for FrequencyClassDescriptorCode. See the domain value list for FrequencyClassType for more information.
33. FrequencyClassInformation's LowerClassBoundValue and UpperClassBoundValue may be required depending on the value provided for FrequencyClassDescriptorCode. See the domain value list for FrequencyClassType for more information

4.2 Delete

The XML Schema for the “Delete” operation is very simple (see Exhibit 2).

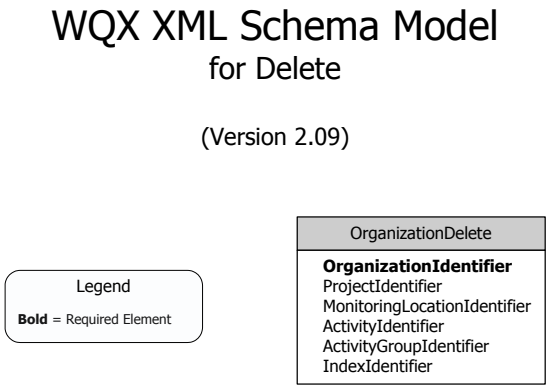


Exhibit 2 – Delete Schema

OrganizationIdentifier is required (and must be provided only once). The OrganizationIdentifier is required for context only. Organizations cannot be deleted.

In addition to the OrganizationIdentifier, at least one of the other identifiers (e.g. ProjectIdentifier, MonitoringLocationIdentifer, ActivityIdentifier, ActivityGroupIdentifier, or IndexIdentifier) must be provided and can be repeated as many times as necessary to identify all of the records to be deleted from the WQX Database.

The following procedure defines the delete process:

The unique identifier for each Project, Monitoring Location, Activity, Biological/Habitat Index and Activity Group in the payload is compared with the identifiers for these respective items in the WQX Database. If the identifier is not found, a warning is added to the Processing Report, and the identifier is ignored. If the identifier is found, then the matching record in the database is deleted.

When one of these primary components is deleted from the system, all child data is deleted as well.

For example:

- AttachedBinaryObjects are deleted whenever the parent Project, MonitoringLocation, Activity or Result is deleted;
- Results are deleted whenever their parent Activity is deleted;
- Activites are deleted whenever their parent Project or MonitoringLocation is deleted.

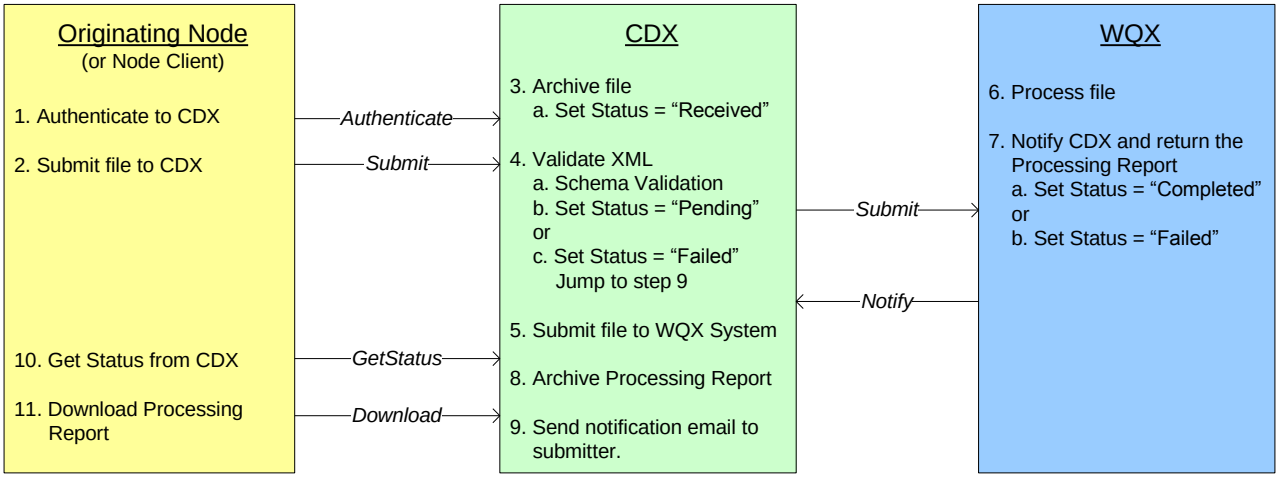
The following exceptions exist:

1. Activities can relate to more than one Project.
    a. When a project is deleted and a related Activity has more than one project relationship, the Activity will not be deleted. Only the relationship between the Project and the Activity will be deleted.
    b. When an Activity relates to only one Project, the Activity will be deleted when its parent Project is deleted.
2. Activities may or may not relate to a Monitoring Location.
    a. When an Activity relates to a Monitoring Location it will be deleted when its parent Monitoring Location is deleted.
    b. When it does not relate to a Monitoring Location (such as for certain Quality Control Samples) it will be unaffected by any Monitoring Location deletes.
3. Deleting a Biological/Habitat Index or Activity Group does not delete the Activities that relate to them.

5.0 Submission Processing and Feedback

Before a WQX submission is made to CDX, a Node User must register with Network Authentication Authorization (NAAS) and obtain a user account. Furthermore, a NAAS policy is established that allows the account to invoke specific methods on the CDX Node for the WQX exchange.

Submission of Exchange Network Documents to the WQX System via the CDX Node follows these processing steps:



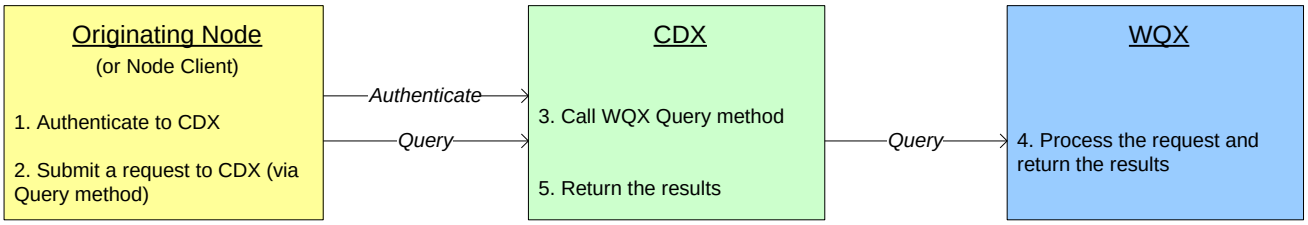
1. The Originating Node/Node Client calls the *Authenticate* method to initiate a session with CDX.
    a. If authentication is successful, a Security Token will be returned.
2. The *Submit* method is called to submit a WQX file for processing.
    a. A Transaction ID is returned, indicating that the file transfer was successful.
3. The submission file (.zip or .xml) is archived at CDX.
    a. The status of the submission is set to "Received".
4. The XML submission file is validated at CDX:
    a. The file is compared with the Header Schema and the relevant WQX Schema to confirm that it complies with the required structure.
    b. If the XML file passes validation, the submission status is set to "Pending".
    c. Otherwise, a Processing Report is generated and the submission status is set to "Failed". No further processing occurs (skip to step 9).
5. CDX submits the file to the WQX System (by calling its *Submit* method).
6. The WQX System processes the submission file (i.e. performs database inserts, updates, and deletes as indicated in the file) and loads any attached binary objects.
7. The WQX System calls the *Notify* method at CDX to update the submission status and return a Processing Report.
    a. If the file was processed without errors, then the status is set to "Completed".
    b. If there were errors, then the status is set to "Failed".
8. The Processing Report is archived at CDX.
9. An email is sent to the submitter notifying him/her of the final status of the submission ("Failed" or "Completed").
10. The Originating Node/Node Client calls the *GetStatus* method to determine the status of the submission. If the Originating Node is manually controlled and the submitter is already aware of the status (from the email in step 9), then this step could be skipped.
11. Once the status is either "Completed" or "Failed", the *Download* method is called to retrieve the Processing Report from CDX.
    a. The Processing Report is an XML-based summary of the processing that occurred on the WQX System (including processing time, insert/update/delete counts and any error or warning messages). See Appendix A for an example Processing Report.

6.0 Query Processing and Feedback

A query allows a State/Regional/Tribal node to request data back from the WQX system. There are two methods for retrieving this data: the Query method, which returns the results immediately, and the Solicit method which creates a data file (offline) which can be downloaded (or submitted back to the requestor's node) at a later time. The Query method is ideal for smaller requests and may have restrictions on the size of the results returned. The Solicit method can accommodate larger requests.

6.1 Query Process

The Query process follows these processing steps:



1. The Originating Node/Node Client calls the Authenticate method to initiate a session with CDX.
    a. If authentication is successful, a Security Token will be returned.
2. The *Query* method is called to request data from CDX.

*3. CDX forwards the request on to the WQX System (via the *Query* method).

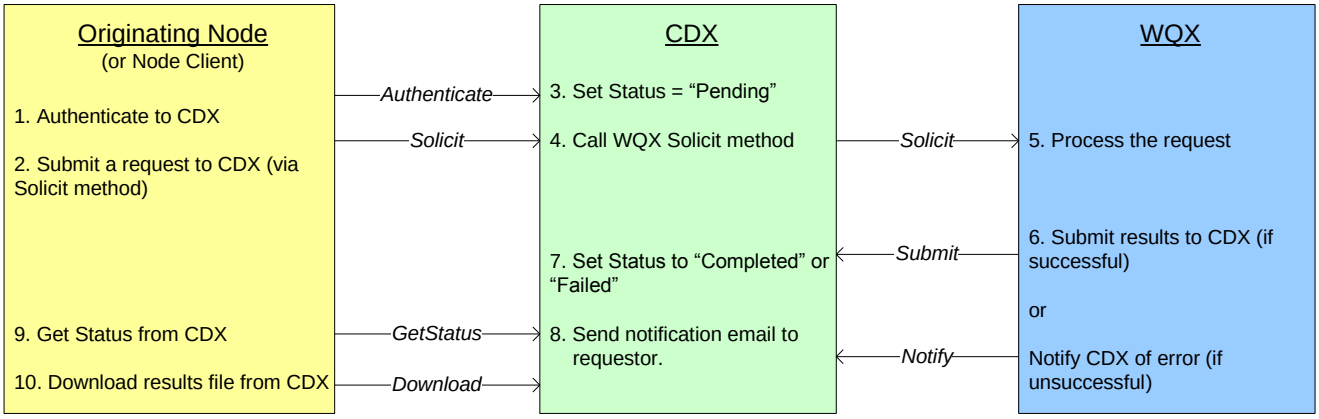
*4. The WQX System processes the request and returns the results to CDX.

*5. CDX returns the results to the Originating Node/Node Client.

* Steps 3, 4, and 5 are all transparent to the Originating Node. The Query method call (in step 2) returns the results (mentioned in step 5).

6.2 Solicit Process

The Solicit process follows these processing steps:



1. The Originating Node/Node Client calls the *Authenticate* method to initiate a session with CDX.
    a. If authentication is successful, a Security Token will be returned.
2. The *Solicit* method is called to request data from CDX.
    a. A Transaction ID is returned.
3. The Transaction Status is set to "Pending".
4. CDX calls the *Solicit* method on the WQX System.
5. The WQX System processes the request and creates a results file corresponding to the criteria it was passed.
6. If the request is successful, then the WQX System calls the *Submit* method to return the results file to CDX.

   If the request fails, then the WQX System calls the *Notify* method to return the error message.

7. CDX sets the Transaction Status to “Completed” if the request was successful or “Failed” if not successful.
8. An email is sent to the requestor notifying him/her of the final status of the solicit request (“Failed” or “Completed”).
9. The Originating Node/Node Client calls the *GetStatus* method to determine the status of the transaction. If the Originating Node is manually controlled and the requestor is already aware of the status (from the email in step 8), then this step could be skipped.
10. Once the status is either “Completed” or “Failed”, the *Download* method is called to retrieve the results file (or error message).

7.0 Web Service Methods at CDX

The following sections discuss the web service methods available at CDX for the WQX data flow. This reference can be used by the developers of a State/Regional/Tribal Node in implementing the interface between their node and CDX. For more information, see the Network Node Functional Specification document available on the Exchange Network web site (www.exchangenetwork.net):

7.1 Authenticate

The authenticate method is used to obtain a security token from the Network Authentication Authorization Service (NAAS). This token will be passed in all subsequent method calls in the exchange between your node and CDX. Please note that security tokens will expire 20 minutes after the token is issued.

Parameters:

- userId: the User ID identifying your node.
- credential: the password you were issued along with the User ID.
- authenticationMethod: the method used to authenticate. Currently only "password" is allowed

Returns: a securityToken used to identify your session with CDX.

7.2 Submit

The Submit method is used to send your WQX submission file to CDX.

Parameters:

- securityToken: A security token issued by the NAAS and returned from the Authenticate method.
- transactionId: unused.
- dataflow: The name of target dataflow: “WQX”
- documents: An array of type nodeDocument. Each nodeDocument structure contains a single submission document/file. For WQX, only a single instance of nodeDocument is submitted. The nodeDocument's "documentName" property must have a file extension of .xml or .zip (e.g. "TestSubmision.xml"). Any documents with other extensions in the documentName property will not be processed.

Returns: a transactionId which identifies your submission. This can be used in the GetStatus and Download methods (described below).

7.3 GetStatus

The GetStatus method is used to check on the current status of a submission file being processed or a solicitation for WQX data. Generally this is used to followup on a previous call to the Submit or Solicit method.

Parameters:

- securityToken: A security token issued by the NAAS and returned from the Authenticate method.
- transactionId: A Transaction ID returned by the Submit or Solicit method.

Returns: the status of the specified transaction: Received, Pending, Completed, or Failed.

7.4 Download

The Download method is used to download documents from the CDX node relating to a specific transactionId. Generally, this method is used after the GetStatus returns either “Completed” or “Failed”.

There are two purposes for the Download method:

1. Retrieve the Processing Report after a submission file has been processed.
    a. This report describes the processing that occurred on your submission and any errors or warnings that arose.
2. Retrieve the results of a request for WQX data using the Solicit method.

Parameters:

- securityToken: A security token issued by the NAAS and returned from the Authenticate method.
- transactionId: A transaction ID returned by the Submit or Solicit method.
- dataflow: The name of the dataflow: "WQX".
- documents: (optional).

  If this parameter is left blank, then all documents relating to this transaction will be returned. This parameter is useful to avoid downloading things you don't need. For example: pass in the Processing Report name to download just the processing report, and avoid downloading your original submission file along with it.

  The documents parameter is made up of the following three fields:

  - name: the name of the document you wish to download.
  - type: the file type (XML, ZIP, OTHER).
  - content: not used.

Returns: a set of documents.

The following documents are available for download:

<u>For a *Submit* transaction</u>:

1. The original submission document.

2a. The Schema Validation Error Report (this will only exist in cases where CDX found a problem with the format of your XML submission file).
(name = "Validation Results", type = "XML")

or

2b. The Processing Report from the WQX System.
(name = "DocumentStatus_ +[original document name]", type = "XML")

<u>For a *Solicit* transaction</u>:

1. The results file.
(name = "Results.zip", type = "ZIP").

7.5 Query

The Query method is used to query the WQX system and retrieve data from it. Queries have restrictions to keep the size of the output reasonably small. For requests that may return a larger set of data, please use the Solicit method. Section 6.6.1 describes the Query and Solicit Requests that are currently support by the WQX system.

Parameters:

- securityToken: A security token issued by the NAAS and returned from the Authenticate method.
- request: The name of the query to be performed (see section 6.6.1 for more details).
- rowId: not used.
- maxRow: not used.
- parameters: An array of parameter values for the query to be performed. Parameters are specific to the request that is made (see section 6.6.1 for details).

Returns: a set of data in an XML instance document. The returned data set is dependant on the request made.

7.6 Solicit

The Solicit method is used to query the WQX system and retrieve data from it. Unlike the Query method, which returns the results immediately, the Solicit method processes the request offline and creates a zipped XML document that can be downloaded (or submitted back to the requestor's node) at a later time. The Solicit method is also able to handle requests for a larger volume of data than the Query method. Section 6.6.1 describes the requests that are currently support by the WQX system.

Parameters:

- securityToken: A security token issued by the NAAS and returned from the Authenticate method.
- returnURL: Not supported at CDX (leave blank).
- request: The name of the query to be performed (see section 6.6.1 for more details).
- parameters: An array of parameter values for the query to be performed. Parameters are specific to the request that is made (see section 6.6.1 for details).

Returns: a transactionId which identifies your request. This can be used with the *GetStatus* and *Download* methods to retrieve your results.

8.0 Data Services (Query/Solicit)

WQX will support the following Requests available via the *Query* or *Solicit* Web Service Methods:

- **WQX.GetActivityByParameters_v2.0 or WQX.GetActivityByParameters_v2.1**
  - Description: Returns a collection of Activities and a count of the number of Results on each Activity.
  - Available via: Query or Solicit
    - Query requests are limited to 500 Activities.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	
2	monitoringLocationIdentifier	String	Optional	
3	projectIdentifier	String	Optional	
4	activityStartDateBegin	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
5	activityStartDateEnd	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
6	activityIdentifier	String	Optional	

  - Return Schema: WQX_WQX_v2.1.xsd (Organization & Activity Sections only).

- **WQX.GetActivityGroupByParameters_v2.0 or WQX.GetActivityGroupByParameters_v2.1**
  - Description: Returns a collection of Activity Groups.
  - Available via: Query or Solicit.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	

2	activityGroupTypeCode	String	Optional	
3	activityGroupIdentifier	String	Optional	

- Return Schema: WQX_WQX_v2.1.xsd (Organization & ActivityGroup Sections only).

- **WQX.GetBiologicalHabitatIndexByParameters_v2.0 or WQX.GetBiologicalHabitatIndexByParameters_v2.1**
  - Description: Returns a collection of Biological or Habitat Indices.
  - Available via: Query or Solicit
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	
2	monitoringLocationIdentifier	String	Optional	
3	indexCalculatedDateBegin	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
4	indexCalculatedDateEnd	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
5	indexIdentifier	String	Optional	

  - Return Schema: WQX_WQX_v2.1.xsd (Organization & BiologicalHabitatIndex Sections only).

- **WQX.GetDomainValueByElementName_v2.0 or WQX.GetDomainValueByElementName_v2.1**
  - Description: Returns a list of domain values for the elementName passed in.
  - Available via: Query or Solicit
    - Query requests are limited to one Domain List (i.e. elementName is required).
  - Parameters:

Pos	Name	Type	Required	Notes
1	elementName	String	Required for Query Optional for Solicit	If the elementName is null, then all domain value lists are returned. Otherwise, elementName must be one of the following: ActivityGroupType ActivityMedia ActivityMediaSubdivision ActivityRelativeDepth ActivityType AddressType AnalyticalMethod AnalyticalMethodContext Assemblage BiologicalIntent CellForm CellShape Characteristic Country County DetectionQuantitationLimitType

				ElectronicAddressType FrequencyClassDescriptor Habit HorizontalCollectionMethod HorizontalCoordinateReferenceSystemDatum MeasureUnit MethodSpeciation MetricType MetricTypeContext MonitoringLocationType NetType ReferenceLocationType ResultDetectionCondition ResultLaboratoryComment ResultMeasureQualifier ResultMeasureValuePickList ResultSampleFraction ResultStatus ResultTemperatureBasis ResultTimeBasis ResultValueType ResultWeightBasis SampleCollectionEquipment SampleContainerColor SampleContainerType SampleTissueAnatomy SamplingDesignType State StatisticalBase Taxon TelephoneNumberType TimeZone ThermalPreservativeUsed ToxicityTestType Tribe VerticalCollectionMethod VerticalCoordinateReferenceSystemDatum Voltinism WellFormationType WellType

- Return Schema: WQX_DomainValues_v2.1.xsd.

- **WQX.GetDomainValueByElementName_v1.0**
  - Description: For backward compatibility with WQX 1.0, the version 1.0 name is still supported.
  - Parameters: identical to the 2.0 version described above.
  - Return Schema: WQX_DomainValues_v1.0.xsd

- **WQX.GetMonitoringLocationByParameters_v2.0 or WQX.GetMonitoringLocationByParameters_v2.1**
  - Description: Returns a collection of Monitoring Locations.
  - Available via: Query or Solicit
    - Query requests are limited to 1000 Monitoring Locations.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	

2	monitoringLocationIdentifier	String	Optional	

- Return Schema: WQX_WQX_v2.1.xsd (Organization & MonitoringLocation Sections only).

- **WQX.GetProjectByParameters_v2.0 or WQX.GetProjectByParameters_v2.1**
  - Description: Returns a collection of Projects for the organization passed in.
  - Available via: Query or Solicit
    - Query requests are limited to 1000 Projects.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	
2	projectIdentifier	String	Optional	

  - Return Schema: WQX_WQX_v2.1.xsd (Organization & Project Sections only).

- **WQX.GetResultByParameters_v2.0 or WQX.GetResultByParameters_v2.1**
  - Description: Returns a collection of Activities and Results.
  - Available via: Query or Solicit.
    - Query requests are limited to 500 Results.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Required	
2	monitoringLocationIdentifier	String	Optional	
3	projectIdentifier	String	Optional	
4	activityStartDateBegin	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
5	activityStartDateEnd	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
6	activityIdentifier	String	Optional	

  - Return Schema: WQX_WQX_v2.1.xsd (Organization, Activity & Results Sections only).

- **WQX.GetTransactionHistoryByParameters_v2.0 or WQX.GetTransactionHistoryByParameters_v2.1**
  - Description: Returns a summary of the transactions processed by the WQX System.
  - Available via: Query or Solicit.
  - Parameters:

Pos	Name	Type	Required	Notes
1	organizationIdentifier	String	Conditionally Required	One of these three parameters must have a non-null value
2	userIdentifier	String	Conditionally Required	

3	transactionIdentifier	String	Conditionally Required	
4	transactionDateBegin	DateTime	Optional	Example format: 2005-10-16T14:00:00-06:00 or 2005-10-16
5	transactionDateEnd	DateTime	Optional	Example format: 2005-10-06T14:00:00-06:00 or 2005-10-16

- Return Schema: WQX_TransactionHistory_v2.1.xsd

- **WQX.GetTransactionHistoryByParameters_v1.0**
  - Description: For backward compatibility with WQX 1.0, the version 1.0 name is still supported.
  - Parameters: identical to the 2.0 version described above.
  - Return Schema: WQX_TransactionHistory_v1.0.xsd

Appendix A. – Example Processing Report


```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="http://www.epa.gov/storet/download/validation.xsl"?>
<ProcessingReport>
  <TransactionIdentifier>6ede652a-6fbc-4af4-9972-37e1d8316ba4</TransactionIdentifier>
  <Status>Failed</Status>
  <ProcessingSoftware Version="2.24">WQX Node</ProcessingSoftware>
  <ProcessingSoftware Version="2.24">WQX Database</ProcessingSoftware>
  <Counts>
    <Error>1</Error>
    <Warning>1</Warning>
    <Project Action="Insert">1</Project>
    <Project Action="Update">0</Project>
    <Project Action="Delete">0</Project>
    <MonitoringLocation Action="Insert">83</MonitoringLocation>
    <MonitoringLocation Action="Update">0</MonitoringLocation>
    <MonitoringLocation Action="Delete">0</MonitoringLocation>
    <Activity Action="Insert">1718</Activity>
    <Activity Action="Update">10</Activity>
    <Activity Action="Delete">0</Activity>
    <ActivityGroup Action="Insert">7</ActivityGroup>
    <ActivityGroup Action="Update">1</ActivityGroup>
    <ActivityGroup Action="Delete">0</ActivityGroup>
    <Result Action="Insert">32021</Result>
    <Result Action="Update">0</Result>
    <Result Action="Delete">0</Result>
    <BiologicalHabitatIndex Action="Insert">21</BiologicalHabitatIndex>
    <BiologicalHabitatIndex Action="Update">0</BiologicalHabitatIndex>
    <BiologicalHabitatIndex Action="Delete">0</BiologicalHabitatIndex>
  </Counts>
  <Log>
    <LogDetail>
      <Type>Message</Type>
      <Text>Parse and Load started at 01/24/2006 03:29:37 PM</Text>
      <Context></Context>
    </LogDetail>
    <LogDetail>
      <Type>Warning</Type>
      <Text>Activity could not be deleted.  Activity ID "25792446" not found.</Text>
      <Context>MonitoringLocationDelete/Line 22, Activity ID = 25792446</Context>
    </LogDetail>
    <LogDetail>
      <Type>Error</Type>
      <Text>File "Station_CBC050.jpg" cannot be found.</Text>
```

```
      <Context>AttachedBinaryObject/Line 274</Context>
    </LogDetail>
    <LogDetail>
      <Type>Message</Type>
      <Text>Parse and Load completed at 01/24/2006 03:45:40 PM</Text>
      <Context></Context>
    </LogDetail>
  </Log>
  <ProcessingFailures>
    <ProjectIdentifier>982</ProjectIdentifier>
    <ProjectIdentifier>678</ProjectIdentifier>
    <MonitoringLocationIdentifier>288892A</MonitoringLocationIdentifier>
    <MonitoringLocationIdentifier>293848X</MonitoringLocationIdentifier>
    <ActivityIdentifier>1928389822</ActivityIdentifier>
    <ActivityIdentifier>3823239822</ActivityIdentifier>
    <ActivityGroupIdentifier>19228</ActivityGroupIdentifier>
    <ActivityGroupIdentifier>19228</ActivityGroupIdentifier>
    <IndexIdentifier>328</IndexIdentifier>
    <IndexIdentifier>367</IndexIdentifier>
  </ProcessingFailures>
</ProcessingReport>
```