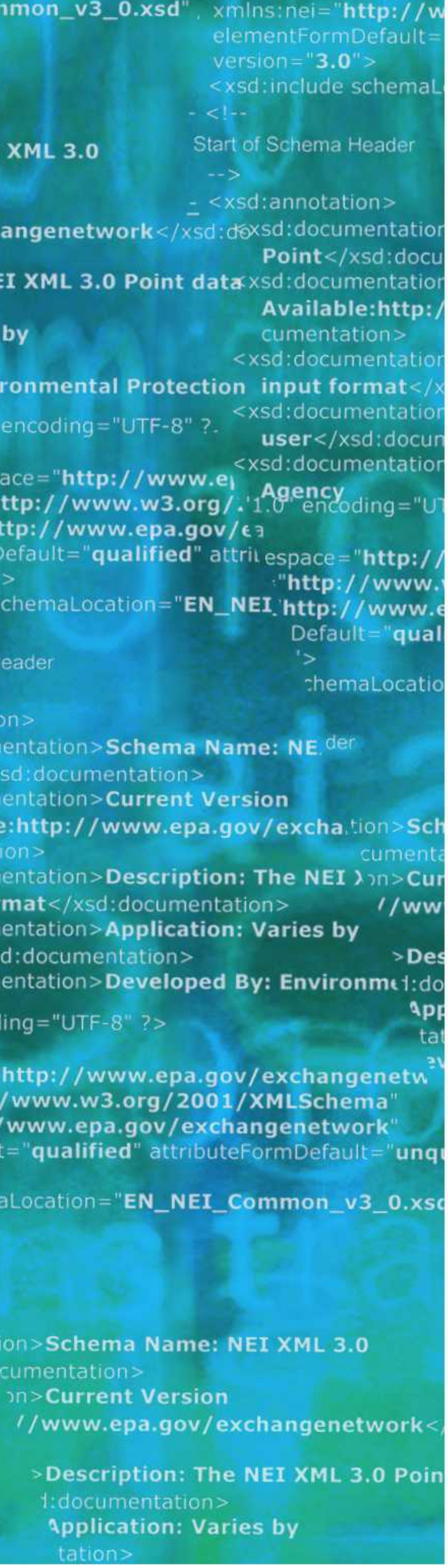


OpenNode2

Beach Notification (BEACHES) 2.x Data Exchange Implementation Guide

Revision Date: 7/9/2014

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
12/15/2009	Windsor	Initial version	1.0
1/28/2011	Windsor	Revision for.NET OpenNode2 v1.2 1. Added support for submission history tracking among other enhancements	2.0
4/11/2012	Windsor	A new plugin has been released to support the Beaches v2.2 exchange. The instructions in this guide are unchanged. The document title has been updated to reflect support for the new schema version.	2.1
10/9/2013	Windsor	Revised cover page	2.2
6/25/2014	Windsor	Added Appendix A: Staging Table Block Diagram	2.3
7/9/2014	Windsor	Updated Install Plugin section to describe pre-bundled plugin process starting with OpenNode2 v2.6	2.4

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

Data Exchange Overview

The purpose of this document is to provide detailed instructions for the installation and configuration of the Exchange Network Beach Notification (BEACHES) data exchange on the .NET implementation of the Exchange Network OpenNode2 (OpenNode2).

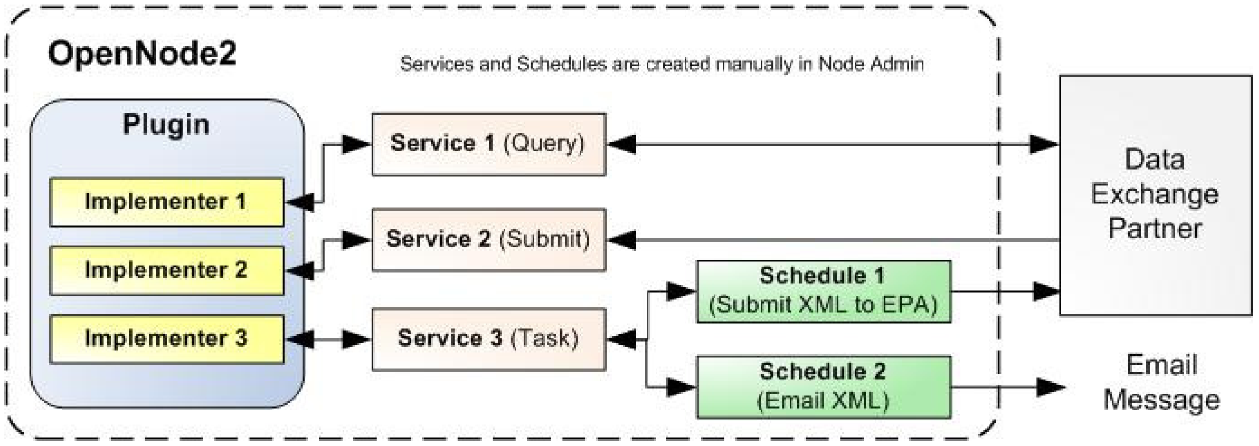
The BEACHES Exchange involves a periodic submittal to EPA for the purposes of updating EPA's PRAWN database with beach advisory data. The frequency and content of the each submission can be configured to meet the agency's needs based on monitoring schedules and data management practices, although EPA only requires BEACH Act grant recipients to submit annually.

The BEACHES data exchange processing workflow can be briefly summarized as follows:

1. Beach notification data is loaded from the State's source database into the OpenNode2 staging database's BEACHES tables. BEACH staging tables are prefixed with "NOTIF_".
2. The BEACHES plugin installed on OpenNode2 will extract BEACHES data from the staging environment, serialize this into an XML document, and submit the resulting XML file to the EPA CDX Node (or another Network partner). The plugin then updates the records in the staging database to indicate that all records have been sent.
3. The EPA CDX Node validates the XML file, and passes it to the internal BEACHES XML processor. If this operation complete, the transaction status is updated to "Completed" by CDX.
4. The EPA BEACHES XML processor validates the contents of the XML file against the PRAWN business rules and, if successful, writes the data to the PRAWN database.
5. EPA CDX sends a notification email to the CDX Web user account, notifying the user of the processing status, and any errors encountered.

Plugin Architecture

The diagram below shows the architecture of a typical OpenNode2 plugin and how services that access the plugin's functionality are configured by a node administrator.



A plugin contains one or more **implementers**. Implementers are canned functionality that are specific to the data exchange. An implementer performs some task, such as composing XML from a series of staging tables.

A node administrator exposes the functionality in an implementer by creating **services**. When a service is created, an implementer must be chosen. Each service may have one or more configuration arguments, defined by the implementer. For example, the service may require that a database connection or node partner URL be provided. Services can be made available to external partners in the form of a query or solicit or as an inbound submission processor. "Task" services are internal only and are accessed via a **schedule**. Schedules also can have configuration arguments which are used by the plugin implementer assigned to the schedule.

BEACHES Plugin Implementers

This section describes the different implementers available in the BEACHES plugin, the arguments they require, and how they operate.

BeachesExtractAndSubmission Implementer

Implementer Name:	**BeachesExtractAndSubmission**
Description/Usage:	This implementer is used for those that wish to use the **Incremental Submission** approach as described in the previous section.

The following steps are performed when this implementer is executed:

1. Searches the NOTIF_SUBMISSIONHISTORY table for any previous transactions that are not yet Completed or Failed. For each, the status is checked at CDX and updated if the status has changed. If any existing transactions are still "pending", executing is halted. This step can be skipped by setting the "UseSubmissionHistory" schedule parameter to "false".
2. Runs a stored procedure to populate/refresh data in the staging

tables. This step can be skipped by leaving the service's "Extract Stored Procedure Name" argument blank.

3. Serialize the data in the staging table to Notification XML. Only retrieve the NOTIF_BEACHADVISORY records where SENTTOEPA is "N".
4. Update the SENTTOEPA flag to "Y" for all records in NOTIF_BEACHADVISORY. This step can be skipped by setting the "UpdateSentToEPAFlag" schedule parameter to "false".
5. Insert a new record into NOTIF_SUBMISSIONHISTORY with the details of the transaction. This step can be skipped by setting the "UseSubmissionHistory" schedule parameter to "false".
6. Subsequent processing is determined by the settings for the schedule that executes this implementer. The XML could be emailed or submitted to another node, for example.

Service Parameters: **Execute Timeout (in seconds):** Sets the timeout for the execution of the data extraction stored procedure.

Extract Stored Procedure Name: Sets the name of the stored procedure that is responsible for refreshing the staging tables with new or changed beach notification data. Leave this parameter blank to skip this step.

Data Source: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: **UseSubmissionHistoryTable**: Set to true or false. If not supplied, the default value is True. Set to false to skip steps 1 and 5 listed above.

UpdateSentToEPAFlag: Set to true or false. If not supplied, the default value is True. Set to false to skip step 4 listed above.

AttributeEffectiveYear: This is the parameter value passed to the Extract Stored Procedure specified in the service parameters. The parameter name is misleading; it can be whatever value need to be passed as an input parameter to the stored procedure.

ClearPendingBeachesSubmissions Implementer

Description/Usage: This implementer will set any "pending" submissions to "failed". This is useful if a submission gets hung up at the receiver end. Executing this implementer will allow the BeachesExtractAndSubmission implementer to continue with submitting data since that implementer will normally abort if "pending" submissions are found.

Service Parameters: **Data Source**: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: None.

ExecuteBeachesExtract Implementer

Description/Usage: This implementer will execute a stored procedure. This service would typically be executed manually from Node Admin to run the stored procedure that loads the staging database with new or changed beach notification data from a source system.

Service Parameters: **Execute Timeout (in seconds):** Sets the timeout for the execution of the data extraction stored procedure.

Extract Stored Procedure Name: Sets the name of the stored procedure that is responsible for refreshing the staging tables with new or changed beach notification data. Leave this parameter blank to skip this step.

Data Source: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: **AttributeEffectiveYear**: This is the parameter value passed to the Extract Stored Procedure specified in the service parameters. The parameter name is misleading; it can be whatever value need to be passed as an input parameter to the stored procedure.

GetBeachesSubmissionStatus Implementer

Description/Usage: This implementer is used for those that wish to use the **Incremental Submission** approach as described in the previous section.

This implementer searches the NOTIF_SUBMISSIONHISTORY table for any previous transactions that are not yet Completed or Failed. For each, the status is checked at CDX and updated if the status has changed.

This implementer should be tied to a service that runs periodically. This will ensure that the status of any previous submission is refreshed in the staging database.

Service Parameters: **Execute Timeout (in seconds):** Sets the timeout for the execution of the data extraction stored procedure.

Extract Stored Procedure Name: Sets the name of the stored procedure that is responsible for refreshing the staging tables with new or changed beach notification data. Leave this parameter blank to skip this step.

Data Source: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: None.

PerformBeachesSubmission Implementer

Description/Usage: This implementer is used for those that wish to use the **Single Submission** approach as described in the previous section.

The following steps are performed when this implementer is executed:

1. Serialize the data in the staging table to Notification XML. Only

retrieve the NOTIF_BEACHADVISORY records where SENTTOEPA is “N”.

2. Submit the XML data to the partner defined in the service’s “Node Partner Name” parameter.
3. Update the SENTTOEPA flag to “Y” for all records in NOTIF_BEACHADVISORY. This step can be skipped by setting the “UpdateSentToEPAFlag” schedule parameter to “false”.
4. Insert a new record into NOTIF_SUBMISSIONHISTORY with the details of the transaction. This step can be skipped by setting the “UseSubmissionHistory” schedule parameter to “false”.

Service Parameters: **Submission Partner Name:** This is the name of the Network Partner that will receive the XML data from OpenNode2. Network Partners are configured on the Configuration tab of Node Admin. Do not put the URL of the partner name in this field as this will not work!

Data Source: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: None.

SubmissionProcessor Implementer

Description/Usage: This implementer is used to process XML data submitted to OpenNode2 by an external partner. The data will then be deserialized to the node staging database, overwriting all existing data. This may useful for testing purposes or if a node needs to be configured to receive BEACHES data from some other partner.

The following steps are performed when this implementer is executed:

1. All existing data is purged from the staging database.
2. The XML in the submission file is deserialized and inserted into the staging database.

To use this implementer, a service must be created with a name of * and a service type of Submit, and set for public access. When a file is submitted to the OpenNode2 endpoint with a dataflow of “Beaches”, this service will run automatically to process the file.

Service Parameters: **Data Destination**: Select the data source for the connection to the staging database where the BEACHES staging tables reside. Data sources are configured on the Configuration tab of Node Admin.

Schedule Parameters: None. This service should not be executed by a schedule.

BEACHES Plugin Configuration Approaches

There are two main approaches to implement the BEACHES exchange using this plugin: Single Submission and Incremental Submission. The node administrator should choose the plugin configuration that are most appropriate for their organization's needs.

Single Submission – This approach is recommended for low-frequency submissions. In this approach, the staging tables are cleaned out and repopulated with all the data for a given submission to EPA. This approach is simplest, wherein the only task to be performed by the plugin is to simply read all the data from the staging tables, formulate the data into XML and submit it to an email address (for review) or to a partner node (for the production submission).

Incremental Submission – This approach is recommended for higher-frequency submissions. Using this approach, new data is incrementally added to the staging tables. Data from prior submissions does not need to be purged from the staging database first. The plugin can be configured to only send what was recently added to the staging environment each time the scheduler runs.

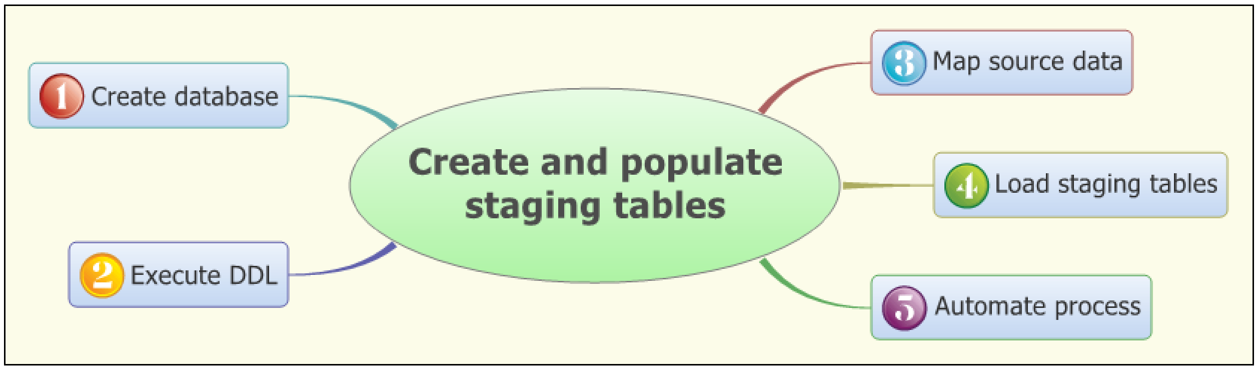
The following sections will list different steps for setting up the plugin services and schedules depending on the desired approach.

Implementing the BEACHES Exchange

Step 1: Create and Populate the BEACHES Staging Tables

OpenNode2 uses a plugin-based architecture to support data exchanges with EPA and other Exchange Network partners. Data must first be loaded into a set of staging tables before it can be extracted by the plugin and shared through the BEACHES data exchange. This section outlines the steps required to set up the BEACHES data exchange database staging tables.

The following figure illustrates these steps:

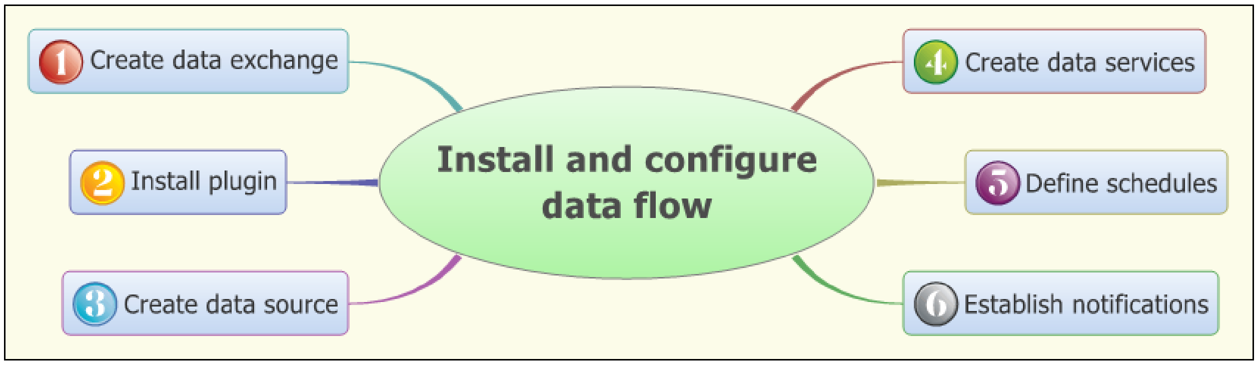


1. The first step is to create the staging database itself if one has not already been established to support another data exchange (typically named NODE_FLOW).
2. Once the staging database is created, a Database Definition Language (DDL) script included in the BEACHES plugin package can be executed to create the staging tables themselves that will be used to store the data being made available through the BEACHES data exchange. Scripts are available for both Oracle and SQL Server.
3. With the staging environment established, data must now be mapped from the source database to the equivalent fields in the BEACHES staging tables. The staging tables closely reflect the structure and naming of the BEACHES XML schema, and it is recommended that the Data Exchange Template (DET) published at exchangenetwork.net be used to facilitate this mapping.
4. Once the mapping is complete, a database routine should be developed to extract the data from the source system into the staging database tables using the mapping prepared during the previous step.
5. Once the data extract process has been developed, it should be automated to execute on a regular schedule as appropriate to the needs of the organization for submissions to EPA. If the extract process is encapsulated by a stored procedure residing within the staging database, the extraction can be executed by the plugin. See the following section for details.

Step 2: Install and Configure the BEACHES Data Exchange

This section describes the steps required to install and configure the BEACHES data exchange on the Microsoft .NET implementation of the OpenNode2 using the Node Administration Web application (Node Admin).

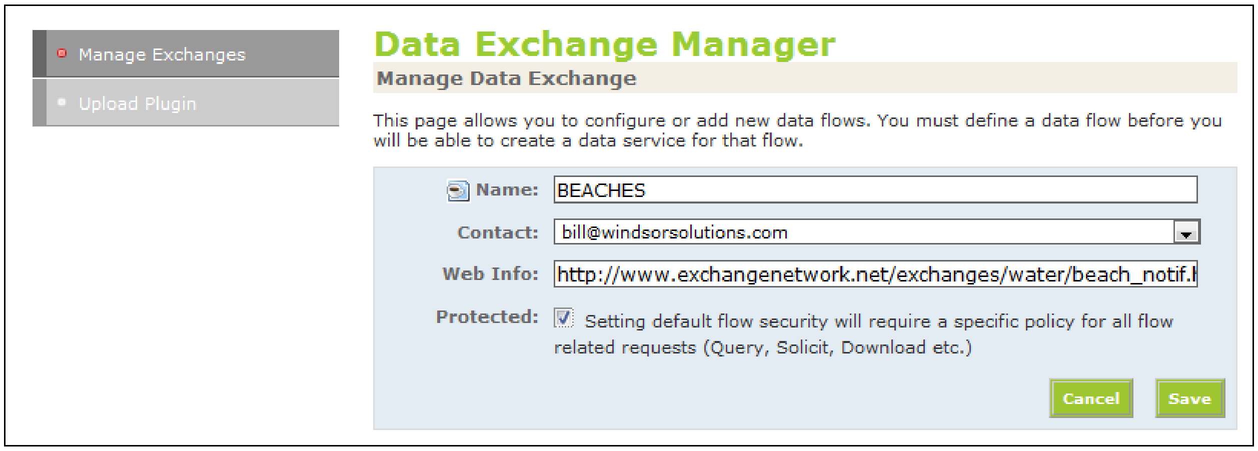
The following figure illustrates these steps:



Create the BEACHES Data Exchange

The first step is to create the BEACHES data exchange using the Node Admin Web application.

1. After logging into the Node Admin, click the **Exchange** tab on the top navigation bar.
2. Click the **Add Exchange** button. The Manage Data Exchange screen will be displayed as follows:



This page allows you to configure or add new data flows. You must define a data flow before you will be able to create a data service for that flow.

Name: BEACHES
Contact: bill@windsorsolutions.com
Web Info: http://www.exchangenetwork.net/exchanges/water/beach_notif.l
Protected: ✓ Setting default flow security will require a specific policy for all flow related requests (Query, Solicit, Download etc.)

3. Type “BEACHES” in the **Name** field.
4. Type a short description in the **Description** field.
5. Select a user account name from the **Contact** drop down box. Contacts are populated with all accounts that have been set up on the OpenNode2. See the **Security** tab for a list of available accounts.

6. In the **Web Info** field, enter a URL where more information can be found about the BEACHES exchange. It is recommended that the following URL be used for this purpose http://www.exchangenetwork.net/exchanges/water/beach_notif.htm.
7. It is recommended that the **Protected** box be checked. This will limit external access to the BEACHES data services. If the node needs to receive BEACHES submissions from external partners, uncheck this box.
8. Click the **Save** button to save the data exchange to OpenNode2.

Install the BEACHES Plugin

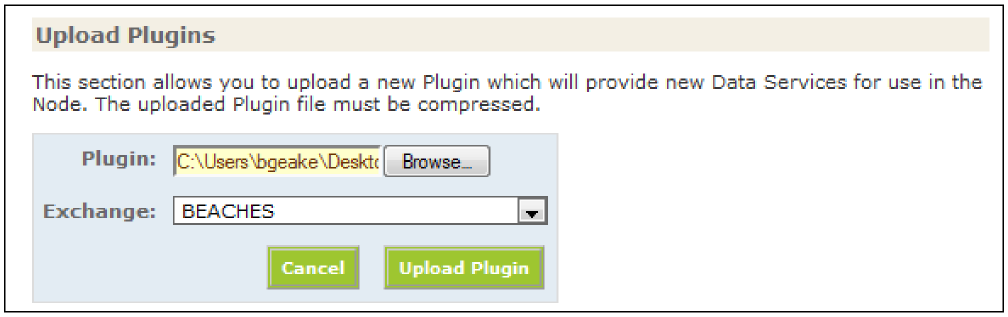
Once the data exchange has been created, the next step is to upload the BEACHES plugin into the OpenNode2 plugin repository.

Note: If you are using OpenNode2 v2.6 or higher, this step is not necessary. Starting with v2.6, all plugins are pre-installed with the OpenNode2 software installation package. By creating the exchange above, the plugin will automatically be loaded and associated with the exchange. To validate that the plugin was installed automatically, follow the steps below:

1. From the **Exchange** tab, scroll down the list of installed data exchanges until the WQX exchange is located.
2. Click the **Add Service** button located just beneath the WQX data exchange record. If the Implementer drop down box is not empty, then the plugin has been installed successfully.

If the steps above reveal that the plugin is not installed, perform the following steps to install it.

1. Navigate to the plugin directory in the **Plugins\[Flow Name]\[version number]** directory included with the OpenNode2 installation files.
2. Create a new zip file containing the two Windsor.Node2008.WNOSPlugin.[Flow name].dll and .pdb files.
3. Click the **Exchange** tab on the top navigation bar.
4. Click the **Upload Plugin** section on the left navigation bar. The Upload Plugin screen will be displayed as follows:


Upload Plugins

This section allows you to upload a new Plugin which will provide new Data Services for use in the Node. The uploaded Plugin file must be compressed.

Plugin: C:\Users\bgeake\Desktc

Exchange: BEACHES

5. Click the **Browse** button which is located to the right of the **Plugin** field.
6. Locate and select the compressed (zipped) file containing the code component for the BEACHES plugin you created in Step 2 above.

7. Select the data exchange name "BEACHES" created during the previous step from the **Exchange** dropdown box.
8. Click the **Upload Plugin** button to upload the plugin.

The newly uploaded plugin code will be placed in the OpenNode2 plugin repository. Any previous plugin versions will be retained in the repository but won't be accessible through the Node Admin. Only the latest version of any one plugin is made available during the next step to establish data services.

Step 3: Create the BEACHES Data Services

The configuration of data services for the BEACHES exchange will depend on the selected implementation approach of either Single Submission or Incremental Submission. See the appropriate section below for the setup that is appropriate for the chosen implementation approach.

Services for Single Submission Approach

If the exchange will be set up for manually executed submissions at infrequent intervals, follow the steps below. Only one data service will need to be established named PerformBeachesSubmission.

Create the PerformBeachesSubmission Data Service

1. From the **Exchange** tab, scroll down the list of installed data exchanges until the BEACHES exchange is located.
2. Click the **Add Service** button located just beneath the BEACHES data exchange record. The following page will be displayed to allow a new data service to be added.
3. In the **Service Name** field, type "PerformBeachesSubmission".
4. From the **Implementer** drop down box, select the value "Windsor.Node2008.WNOSplugin.BEACHES_21. PerformBeachesSubmission"

   Note: When the implementer is selected, several arguments and data sources will appear. The OpenNode2 application will obtain these properties directly from the Beaches plugin.
5. From the **Type** drop down box, select how you wish to make the services available. The options available will be obtained from the plugin. Select "Task".
6. Enable the service by clicking the **Active** checkbox.
7. Based on the selection made from the implementer dropdown box, OpenNode2 will determine what argument and data source requirements the plugin has and will refresh the page to display the relevant data entry fields as follows:
8. Arguments:
   a. Submission Partner Name: This is the name of the Network Partner that will receive the XML data from OpenNode2. Network Partners are configured on the Configuration tab of Node Admin.
9. Data Sources:
   b. Data Source: Select the data source where the beaches staging data resides. Data sources are maintained on the Configuration tab of node admin.
10. Click the **Save** button to save the service.

Skip forward to the *Define Data Exchange Schedules* section for next steps.

Services for Incremental Submission Approach

If the exchange will be set up to send incremental beach notification data to EPA, follow the steps below. Three data services will be established:

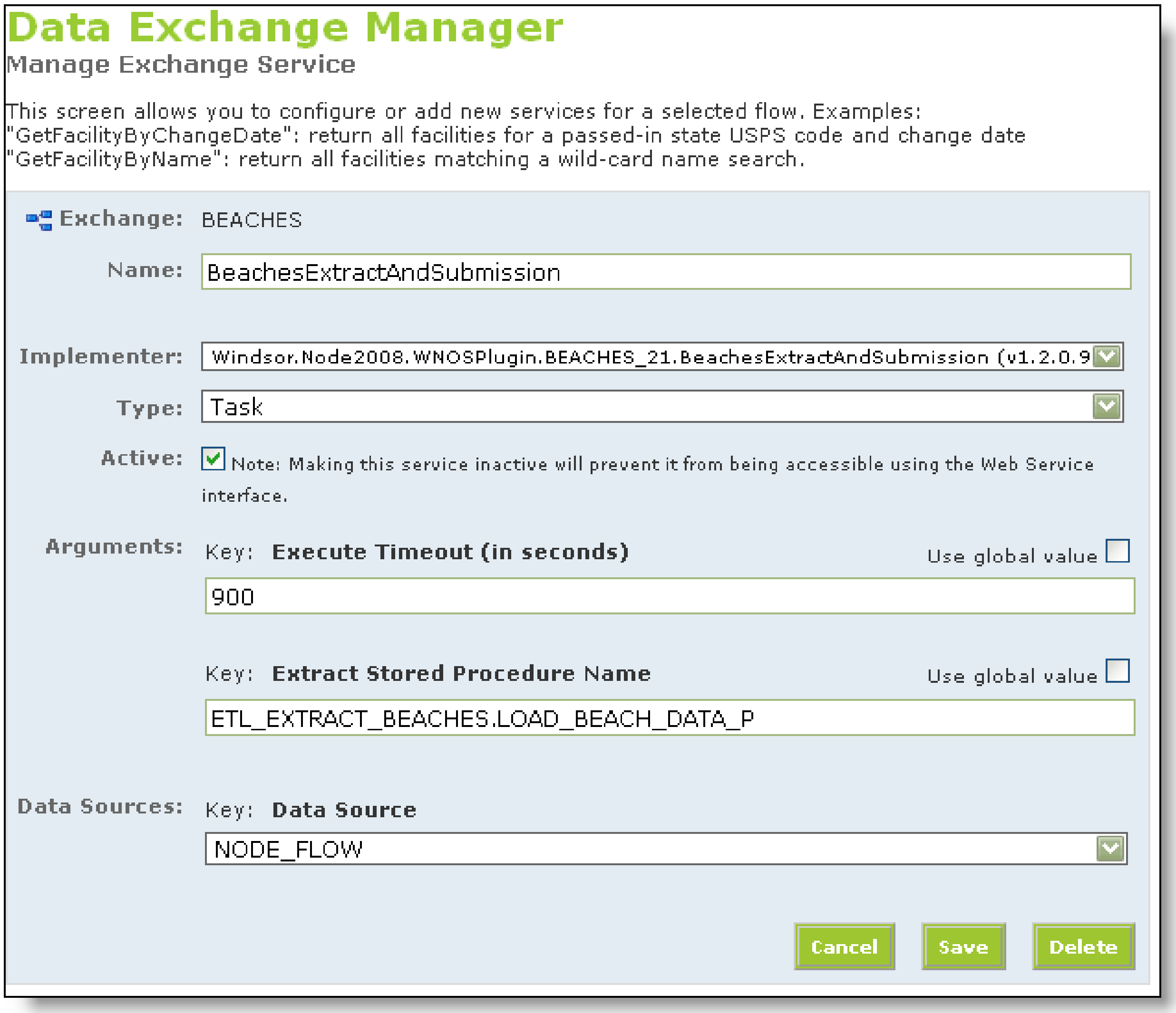
- BeachesExtractAndSubmission
- GetBeachesSubmissionStatus
- ClearPendingBeachesSubmission

Each of these data services must be created and configured before they can be executed.

Create the BeachesExtractAndSubmission Data Service

This data service creates an XML file from data in the staging database. The service returns the XML document to the caller, which is typically a Schedule. The data service can also be configured to execute the data extraction stored procedure prior to generating the XML, if the implementer wishes to use it.

11. From the **Exchange** tab, scroll down the list of installed data exchanges until the BEACHES exchange is located.
12. Click the **Add Service** button located just beneath the BEACHES data exchange record. The following page will be displayed to allow a new data service to be added.



This screen allows you to configure or add new services for a selected flow. Examples:
"GetFacilityByChangeDate": return all facilities for a passed-in state USPS code and change date
"GetFacilityByName": return all facilities matching a wild-card name search.

Exchange:	BEACHES
Name:	BeachesExtractAndSubmission
Implementer:	Windsor.Node2008.WNOSPlugin.BEACHES_21.BeachesExtractAndSubmission (v1.2.0.9
Type:	Task
Active:	✓ Note: Making this service inactive will prevent it from being accessible using the Web Service interface.
Arguments:	Key: Execute Timeout (in seconds) — Use global value ☐ — 900
	Key: Extract Stored Procedure Name — Use global value ☐ — ETL_EXTRACT_BEACHES.LOAD_BEACH_DATA_P
Data Sources:	Key: Data Source — NODE_FLOW

13. In the **Service Name** field, type “BeachesExtractAndSubmission”.
14. From the **Implementer** drop down box, select the value “Windsor.Node2008.WNOSplugin.BEACHES_21.BeachesExtractAndSubmission”

    Note: When the implementer is selected, several arguments and data sources will appear. The OpenNode2 application will obtain these properties directly from the Beaches plugin loaded previously.
15. From the **Type** drop down box, select how you wish to make the services available. The options available will be obtained from the plugin. Select “Task”.
16. Enable the service by clicking the **Active** checkbox.
17. Based on the selection made from the implementer dropdown box, OpenNode2 will determine what argument and data source requirements the plugin has and will refresh the page to display the relevant data entry fields as follows:
18. Arguments:

    c. Execute Timeout: “900”

d. Extract Stored Procedure Name: Enter the name of the extraction stored procedure, if one has been created. To bypass this step, leave the value blank.

19. Data Sources:

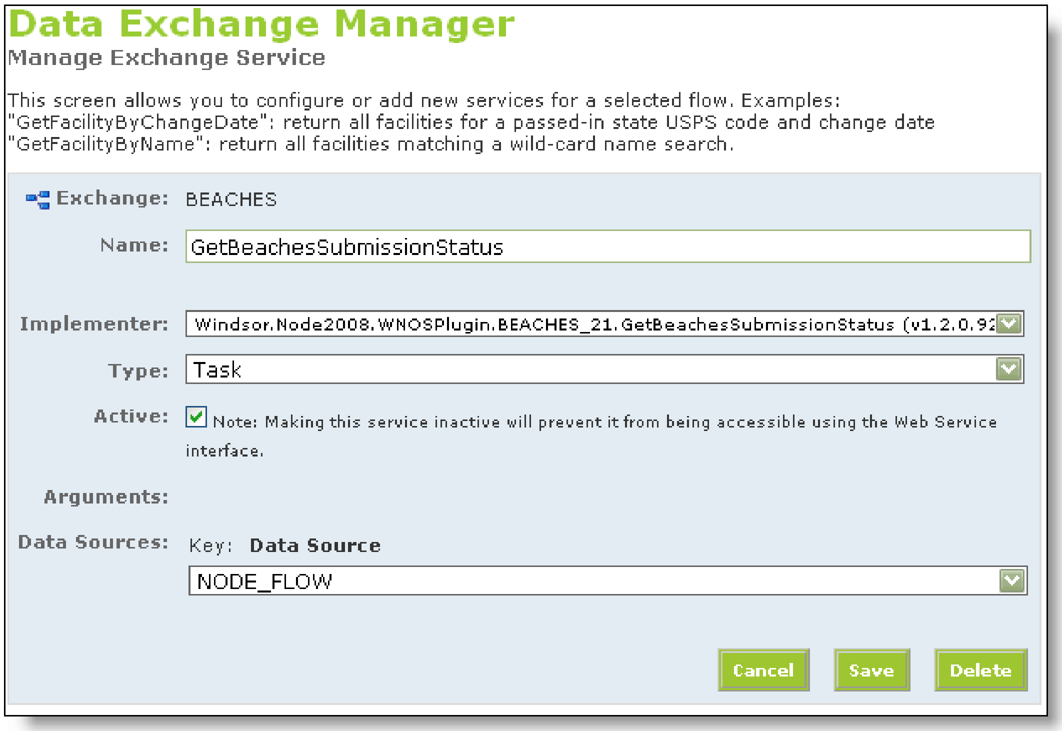
    e. Data Source: Select the data source where the beaches staging data resides. Data sources are maintained on the Configuration tab of node admin.

20. Click the **Save** button to save the service.

Create the GetBeachesSubmissionStatus Data Service

This data service will comb the NOTIF_SUBMISSIONHISTORY table and for each transaction that is not "completed" or "failed", will query the receiving node (typically CDX) for the latest status of the submission and update the table with the latest statuses.

1. From the **Exchange** tab, scroll down the list of installed data exchanges until the BEACHES exchange is located.
2. Click the **Add Service** button located just beneath the BEACHES data exchange record. The following page will be displayed to allow a new data service to be added.



Data Exchange Manager
Manage Exchange Service

This screen allows you to configure or add new services for a selected flow. Examples:
"GetFacilityByChangeDate": return all facilities for a passed-in state USPS code and change date
"GetFacilityByName": return all facilities matching a wild-card name search.

Exchange: BEACHES
Name: GetBeachesSubmissionStatus
Implementer: Windsor.Node2008.WNOSPlugin.BEACHES_21.GetBeachesSubmissionStatus (v1.2.0.9
Type: Task
Active: ✓ Note: Making this service inactive will prevent it from being accessible using the Web Service interface.
Arguments:
Data Sources: Key: Data Source
NODE_FLOW

3. In the **Service Name** field, type "GetBeachesSubmissionStatus".
4. From the **Implementer** drop down box, select the value "Windsor.Node2008.WNOSplugin.BEACHES_21.GetBeachesSubmissionStatus"

   Note: When the implementer is selected, several arguments and data sources will appear. The OpenNode2 application will obtain these properties directly from the Beaches plugin.

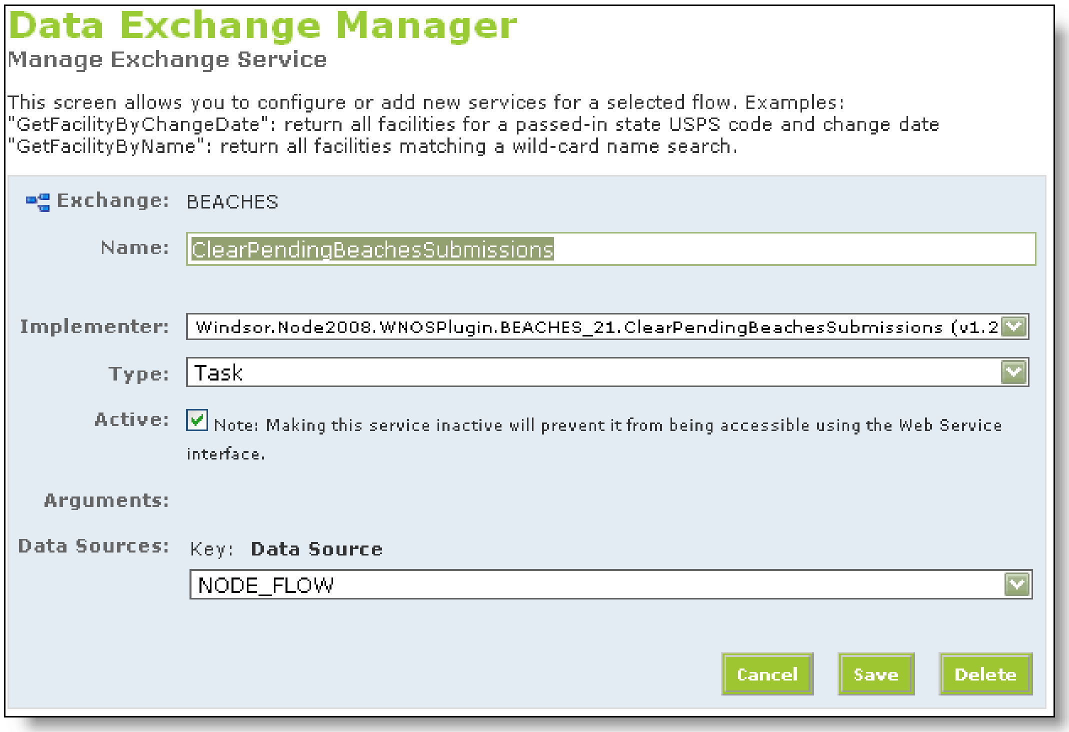
5. From the **Type** drop down box, select how you wish to make the services available. The options available will be obtained from the plugin. Select "Task".

6. Enable the service by clicking the **Active** checkbox.
7. Based on the selection made from the implementer dropdown box, OpenNode2 will determine what argument and data source requirements the plugin has and will refresh the page to display the relevant data entry fields as follows:
8. Data Sources:
   a. Data Source: Select the data source where the beaches staging data resides. Data sources are maintained on the Configuration tab of node admin.
9. Click the **Save** button to save the service.

Create the ClearPendingBeachesSubmission Data Service

This data service will set all "Pending" statuses located in the submission history table (NOTIF_SUBMISSION. PROCESSINGSTATUS) to "Failed". This will only be used if there are one or more transactions that are locked or "hung up" in a pending status. This may occur if the endpoint is having trouble providing back a "Completed" or "Failed" status.

1. From the **Exchange** tab, scroll down the list of installed data exchanges until the BEACHES exchange is located.
2. Click the **Add Service** button located just beneath the BEACHES data exchange record. The following page will be displayed to allow a new data service to be added.



Data Exchange Manager

Manage Exchange Service

This screen allows you to configure or add new services for a selected flow. Examples:
"GetFacilityByChangeDate": return all facilities for a passed-in state USPS code and change date
"GetFacilityByName": return all facilities matching a wild-card name search.

Exchange: BEACHES

Name: ClearPendingBeachesSubmissions

Implementer: Windsor.Node2008.WNOSPlugin.BEACHES_21.ClearPendingBeachesSubmissions (v1.2

Type: Task

Active: ✓ Note: Making this service inactive will prevent it from being accessible using the Web Service interface.

Arguments:

Data Sources: Key: Data Source

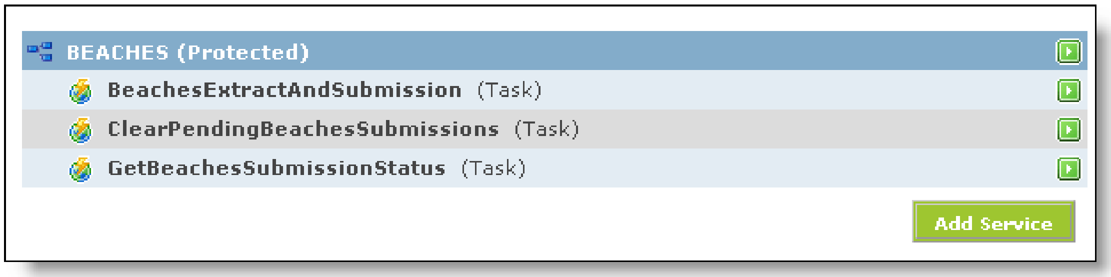
NODE_FLOW

3. In the **Service Name** field, type "ClearPendingBeachesSubmissions".
4. From the **Implementer** drop down box, select the value "Windsor.Node2008.WNOSplugin.BEACHES_21.ClearPendingBeachesSubmission"

   a. *Note: When the implementer is selected, several arguments and data sources will appear. The OpenNode2 application will obtain these properties directly from the Beaches plugin loaded previously.*

5. From the **Type** drop down box, select how you wish to make the services available. The options available will be obtained from the plugin. Select "Task".
6. Enable the service by clicking the **Active** checkbox.
7. Based on the selection made from the implementer dropdown box, OpenNode2 will determine what argument and data source requirements the plugin has and will refresh the page to display the relevant data entry fields as follows:
8. Data Sources:
   a. Data Source: Select the data source where the beaches staging data resides. Data sources are maintained on the Configuration tab of node admin.
9. Click the **Save** button to save the service.

Once completed setting up each data service, the **Manage Exchanges** page for the Beach Notifications data exchange should appear as follows:



BEACHES (Protected)

BeachesExtractAndSubmission (Task)

ClearPendingBeachesSubmissions (Task)

GetBeachesSubmissionStatus (Task)

Step 4: Create Data Exchange Schedules

Scheduled jobs can be configured in the OpenNode2 to perform automated tasks such as submitting data to external partners or processing received files.

The configuration of schedules for the BEACHES exchange will depend on the selected implementation approach of either Single Submission or Incremental Submission. See the appropriate section below for the setup that is appropriate for the chosen implementation approach.

Services for Single Submission Approach

Only one schedule is needed to implement the Single Submission approach, described below.

Create the PerformBeachesSubmission Schedule

1. From the **Schedules** tab, click the **Add Schedule** button.
2. Type "PerformBeachesSubmission" in the **Name** field.
3. Enable the schedule by clicking the **Active** checkbox.
4. Select "BEACHES" from the **Exchange** dropdown list.
5. Set the start date to the first date when you wish the schedule to run. Set the end date to some point after the start date.

Note: If the intention is to manually execute the schedule, the start and end date will not matter.

6. Set the frequency to the data submission to "Once".
7. In the **Data Source** area, check the radio button labeled **Results of local service execution**.
8. In the **Service** dropdown box, select the value "**PerformBeachesSubmission**". This informs the schedule to use the selected BEACHES service as the data source for the submission.
9. In the **Additional Parameters** area, select "By Name" radio button. Add the following parameters. For more information on these parameters see the *BEACHES Plugin Implementers* section of this document.
    a. Name = UpdateSentToEPAFlag. Value = True or False.
    b. Name = UseSubmissionHistoryTable. Value = True or False.
10. In the **Result Process** area, check the radio button for **None.** The plugin will perform the submission to the node endpoint defined in the service. Therefore, the schedule does not need to be set up with a specific target node.

    *Note: For testing purposes, you may set the Result Process to **Send compressed result as an Email Attachment** and specify your own email address as the recipient. This will enable you to manually check the file before submitting.*
11. Click the **Save** button to save the schedule.

Services for Incremental Submission Approach

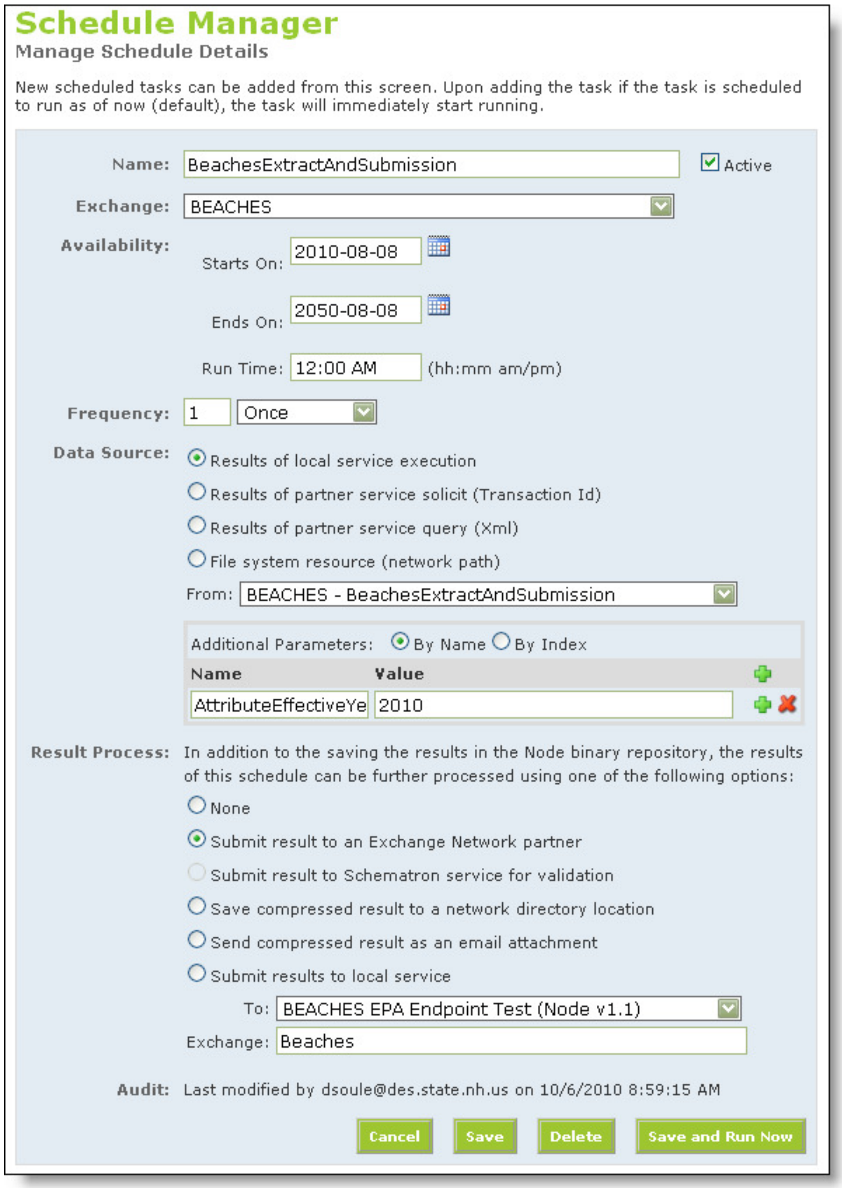
Two schedules will be created:

1. **BeachesExtractAndSubmission** – responsible for creating the XML file and sending it to EPA
2. **GetBeachesSubmissionStatus** – Polls CDX for the status of the submission in the preceding schedule.

The process for creating each schedule is described below:

Create BeachesExtractAndSubmission Schedule

1. From the **Schedules** tab, click the **Add Schedule** button. The Schedule Manager screen will display as follows:



New scheduled tasks can be added from this screen. Upon adding the task if the task is scheduled to run as of now (default), the task will immediately start running.

Name:	BeachesExtractAndSubmission [✓ Active]
Exchange:	BEACHES
Availability:	Starts On: 2010-08-08
	Ends On: 2050-08-08
	Run Time: 12:00 AM (hh:mm am/pm)
Frequency:	1 Once
Data Source:	[● Results of local service execution]
	○ Results of partner service solicit (Transaction Id)
	○ Results of partner service query (Xml)
	○ File system resource (network path)
	From: BEACHES - BeachesExtractAndSubmission
	Additional Parameters: [● By Name] ○ By Index
	Name: AttributeEffectiveYe — Value: 2010
Result Process:	In addition to the saving the results in the Node binary repository, the results of this schedule can be further processed using one of the following options:
	○ None
	[● Submit result to an Exchange Network partner]
	○ Submit result to Schematron service for validation
	○ Save compressed result to a network directory location
	○ Send compressed result as an email attachment
	○ Submit results to local service
	To: BEACHES EPA Endpoint Test (Node v1.1)
	Exchange: Beaches
Audit:	Last modified by dsoule@des.state.nh.us on 10/6/2010 8:59:15 AM

2. Type “BeachesExtractAndSubmission” in the **Name** field.
3. Enable the schedule by clicking the **Active** checkbox.
4. Select “BEACHES” from the **Exchange** dropdown list.
5. Set the desired Start and End Date range in which the schedule will be run.
6. Set the Frequency to the desired interval, such as **1/annual** for annual submissions or **3/months** for quarterly submissions.
7. In the **Data Source** area, check the radio button labeled **Results of local service execution**.
8. In the **Data Source** “from” dropdown box, select the value “BeachExtractAndSubmission.”
9. In the **Additional Parameters** area, select “By Name” radio button. Add the following parameters. For more information on these parameters see the *BEACHES Plugin Implementers* section of this document.

a. Name = UpdateSentToEPAFlag. Value = True or False.

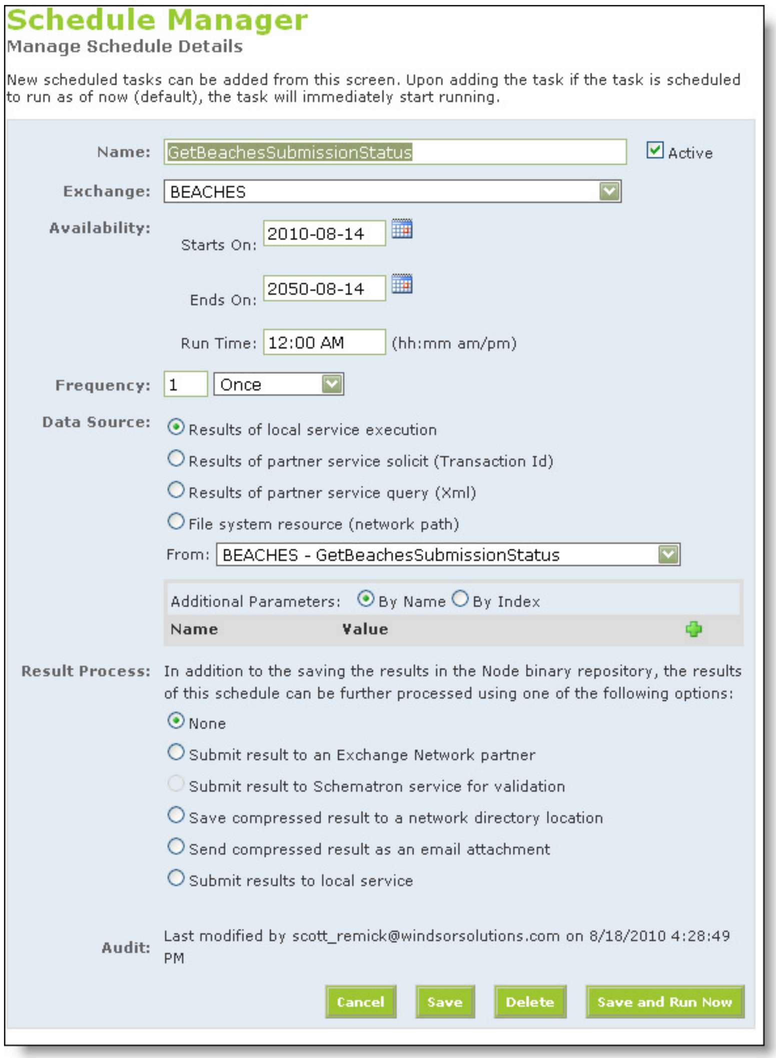
b. Name = UseSubmissionHistoryTable. Value = True or False.

c. Name = AttributeEffectiveYear. Value = (implementation specific)

10. In the **Result Process** area, check the "Submit result to an Exchange Network partner." A new **To**: dropdown will appear with a list of valid Network Partners. Select the appropriate EPA Endpoint". See the Exchange Network Wiki page for a list of appropriate CDX endpoints for the BEACHES exchange.
11. In the **Exchange** field, type in **"**Beaches". The endpoint can be changed by the user depending during each submission.
12. Click the **Save** button to save the schedule. (Note: to run immediately, click the **Save and Run Now** button)

Create GetBeachesSubmissionStatus Schedule

1. From the **Schedules** tab, click the **Add Schedule** button. The Schedule Manager screen will display as follows:



Schedule Manager
Manage Schedule Details

New scheduled tasks can be added from this screen. Upon adding the task if the task is scheduled to run as of now (default), the task will immediately start running.

Name: GetBeachesSubmissionStatus [✓ Active]

Exchange: BEACHES

Availability:
Starts On: 2010-08-14
Ends On: 2050-08-14
Run Time: 12:00 AM (hh:mm am/pm)

Frequency: 1 Once

Data Source:
[✓ Results of local service execution]
Results of partner service solicit (Transaction Id)
Results of partner service query (Xml)
File system resource (network path)
From: BEACHES - GetBeachesSubmissionStatus

Additional Parameters: [✓ By Name] By Index

Name	Value

Result Process: In addition to the saving the results in the Node binary repository, the results of this schedule can be further processed using one of the following options:
[✓ None]
Submit result to an Exchange Network partner
Submit result to Schematron service for validation
Save compressed result to a network directory location
Send compressed result as an email attachment
Submit results to local service

Audit: Last modified by scott_remick@windsorsolutions.com on 8/18/2010 4:28:49 PM

2. Type “GetBeachesSubmissionStatus” in the **Name** field.
3. Enable the schedule by clicking the **Active** checkbox.
4. Select “BEACHES” from the **Exchange** dropdown list.
5. This is a manual flow, and so the **Availability** and **Frequency** can be ignored. Although to “save and now”, the frequency “once” should be selected.
6. In the **Data Source** area, check the radio button labeled **Results of local service execution**.
7. In the **Data Source** “from” dropdown box, select the value “GetBeachesSubmissionStatus.”
8. In the **Result Process** area, check the “None.”
9. Click the **Save** button to save the schedule. (Note: to run immediately, click the **Save and Run Now** button)

The BEACHES schedule is now set up correctly to manage the data flow. Please see the OpenNode2 Administration User Guide for more information on scheduling data exchanges.

Step 5: Contact CDX to Establish Exchange Settings

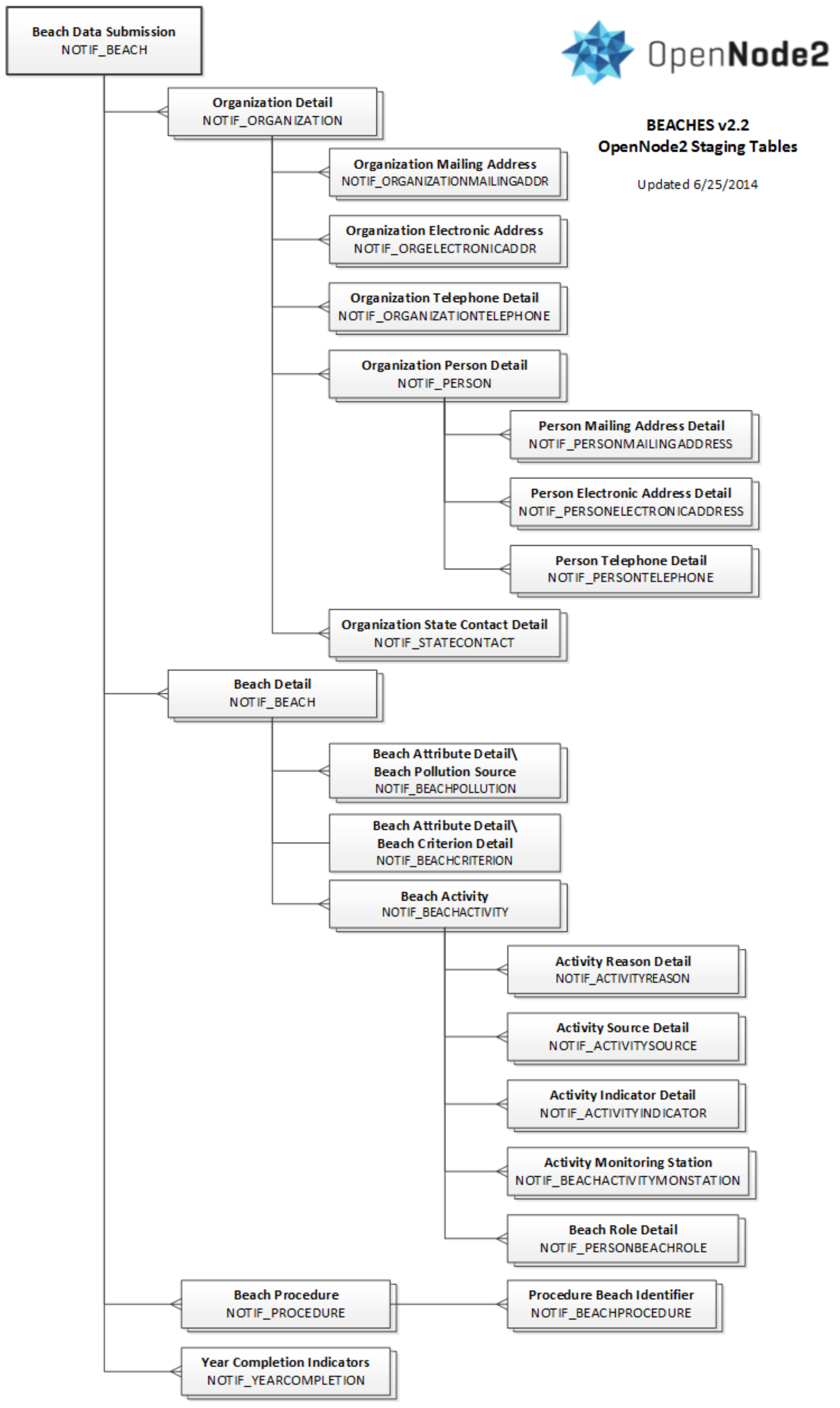
Contact the EPA CDX Node helpdesk and ask them to perform the following tasks:

1. Authorize the OpenNode2 runtime (operator) NAAS account to submit to the BEACHES data exchange on the EPA systems.
2. Map the OpenNode2 runtime NAAS account to the CDX Web user account that currently administers EPA BEACHES data for the organization. This is required to ensure that the EPA-generated emails are sent to the appropriate person in your organization.

Step 6: Establish Email Notifications

If desired, the Node administrator may create NAAS accounts for one or more staff members and create notifications for the any OpenNode2 events related to the BEACHES data exchange. Please see the OpenNode2 Administration User Guide for more information on setting up notifications.

Appendix A: Staging Table Block Diagram


Beach Data Submission
NOTIF_BEACH
OpenNode2
BEACHES v2.2
OpenNode2 Staging Tables
Updated 6/25/2014
Organization Detail
NOTIF_ORGANIZATION
Organization Mailing Address
NOTIF_ORGANIZATIONMAILINGADDR
Organization Electronic Address
NOTIF_ORGELECTRONICADDR
Organization Telephone Detail
NOTIF_ORGANIZATIONTELEPHONE
Organization Person Detail
NOTIF_PERSON
Person Mailing Address Detail
NOTIF_PERSONMAILINGADDRESS
Person Electronic Address Detail
NOTIF_PERSONELECTRONICADDRESS
Person Telephone Detail
NOTIF_PERSONTELEPHONE
Organization State Contact Detail
NOTIF_STATECONTACT
Beach Detail
NOTIF_BEACH
Beach Attribute Detail\ Beach Pollution Source
NOTIF_BEACHPOLLUTION
Beach Attribute Detail\ Beach Criterion Detail
NOTIF_BEACHCRITERION
Beach Activity
NOTIF_BEACHACTIVITY
Activity Reason Detail
NOTIF_ACTIVITYREASON
Activity Source Detail
NOTIF_ACTIVITYSOURCE
Activity Indicator Detail
NOTIF_ACTIVITYINDICATOR
Activity Monitoring Station
NOTIF_BEACHACTIVITYMONSTATION
Beach Role Detail
NOTIF_PERSONBEACHROLE
Beach Procedure
NOTIF_PROCEDURE
Procedure Beach Identifier
NOTIF_BEACHPROCEDURE
Year Completion Indicators
NOTIF_YEARCOMPLETION