



Revision Date: 12/15/2017



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



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
12/15/2015	Windsor	Updated the plugin implementers section with a description of the GetNotificationDetailsByParameters_v2.2 service.	2.5
12/15/2017	Windsor	Update version to v2.4 to reflect plugin schema version. No functional changes added.	2.6

Table of Contents

DATA EXCHANGE OVERVIEW	1
PLUGIN ARCHITECTURE	2
<i>BEACHES Plugin Implementers</i>	3
IMPLEMENTING THE BEACHES EXCHANGE	7
<i>Step 1: Create and Populate the BEACHES Staging Tables</i>	7
<i>Step 2: Install and Configure the BEACHES Data Exchange</i>	8
<i>Step 3: Create the BEACHES Data Services</i>	10
<i>Step 4: Create Data Exchange Schedules</i>	14
<i>Step 5: Contact CDX to Establish Exchange Settings</i>	19
<i>Step 6: Establish Email Notifications</i>	19
APPENDIX A: STAGING TABLE BLOCK DIAGRAM	20

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).

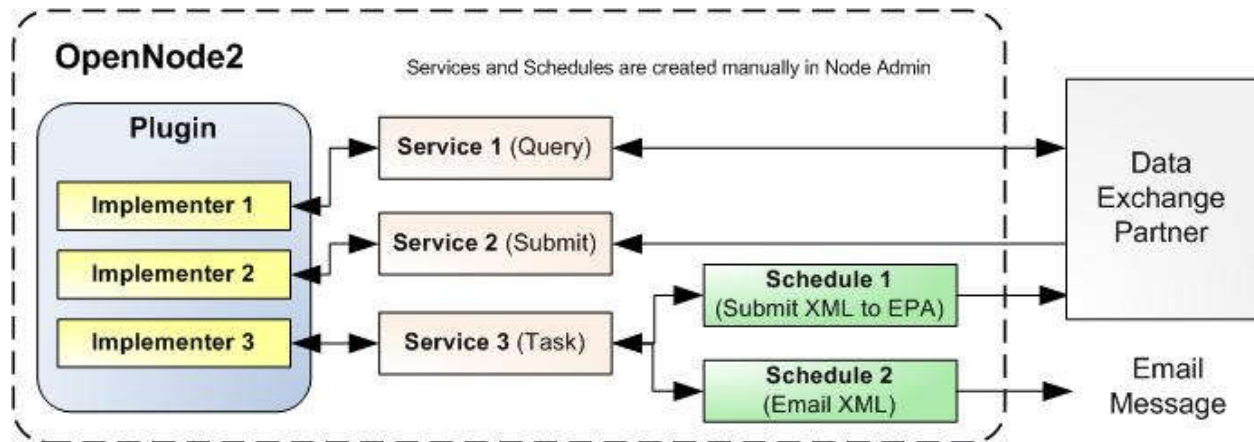
The BEACHES Exchange involves a periodic submittal to EPA for the purposes of updating EPA's PRAWN database with beach advisory data. The frequency and content of the each submission can be configured to meet the agency's needs based on monitoring schedules and data management practices, although EPA 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 database, serialize this into an XML document, and submit the resulting XML file to the EPA CDX Node (or another Network partner).
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 programmed 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 configuring **services**. When a service is created via the OpenNode2 Admin interface, 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.

Incremental Data Submissions

Using the BEACHES OpenNode2 plugin it is possible to submit only new advisories ("activities") to EPA. New activities are defined as those activities that have not previously been successfully submitted.

Activities are stored in the NOTIF_BEACHACTIVITY table. New activities can be indicated in one of two ways:

1. Using the SENTTOEPA column to indicate whether the advisory has been previously sent. Previously submitted activities should be set to Y either manually or by configuring the plugin services to set this flag after submission.
2. Using the NOTIFUPDATEDATE column to indicate when the activity was loaded into the staging table. The plugin can be configured to send only activities that have a NOTIFUPDATEDATE greater than the date of the last successful submission to EPA.

Depending how the plugin is configured, either field can be used to achieve the same result. The second option was added to support a "Delete and Reload" approach to refreshing the staging tables from a source system using an Extract, Transform, and Load (ETL) procedure. Using this method, the ETL does not need to retain information about what was previously sent.

The NOTIF_SUBMISSIONHISTORY table is also an important part of the plugin's ability to send incremental submissions. The plugin can be configured to use this table to record each submittal to EPA.

The plugin can compare the SCHEDULERUNDATE of the most recent successful submission to the NOTIUPDATEDDATE in NOTIF_BEACHACTIVITY to determine which new advisories to send

NOTE: EPA's system does not provide an automated mechanism to update or delete previously sent activities. Corrections to submitted activities must be manually coordinated through EPA's contractor. For this reason, it is important to send activity data that is known to be correct.

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: The following steps are performed when this implementer is executed:

1. If schedule parameter **UseSubmissionHistoryTable** = true, the plugin 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", execution is halted.
1. Execute 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.
2. Serialize the data in the staging table to Notification XML. Only retrieve the NOTIF_BEACHADVISORY records where SENTTOEPA is "N".
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 "UseSubmissionHistoryTable" schedule parameter to "false".
5. 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 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.

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 be useful for testing purposes or if a node needs to be configured to receive BEACHES data from some other partner.

NOTE: EPA does not offer outbound services to retrieve BEACHES data

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.

PerformBeachesSubmission Implementer

Description/Usage: Similar to the BeachesExtractAndSubmission implementer, with two distinctions:

- This service does not allow for calling an ETL procedure prior to creation of XML submission
- This service

Submits Beach notification data to an exchange network partner using one of the two methods described in the Incremental Data Submission section above.

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! If you leave this field blank, the plugin will not track the submission in the submission history table.

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.

GetNotificationDetailsByParameters Implementer

Description/Usage: This service is used to return a list of beach notifications and beach details based on filter criteria supplied by the user. This service allows an external client to inquire for a variety of criteria, such as a list of advisories in a particular date range or a list of current, ongoing advisories.

Service Parameters: Refer to Appendix B for a table of the parameters.

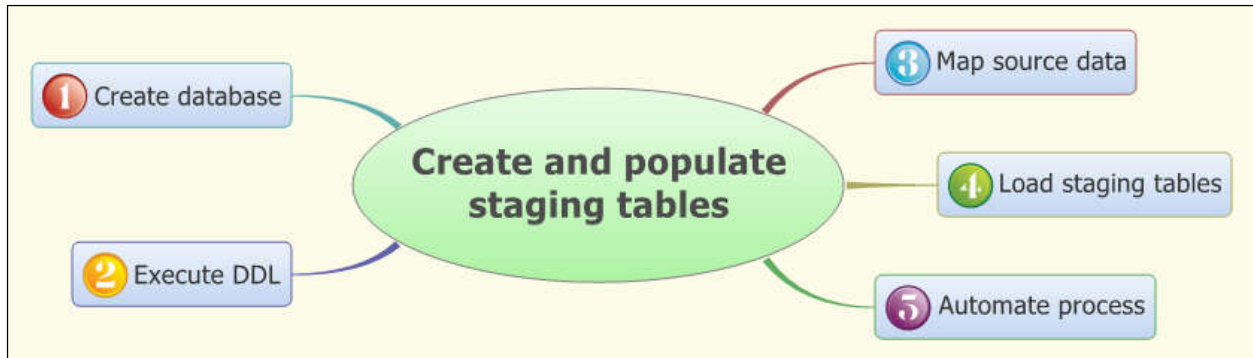
Schedule Parameters: None. This service is not meant to be executed by a schedule.

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:

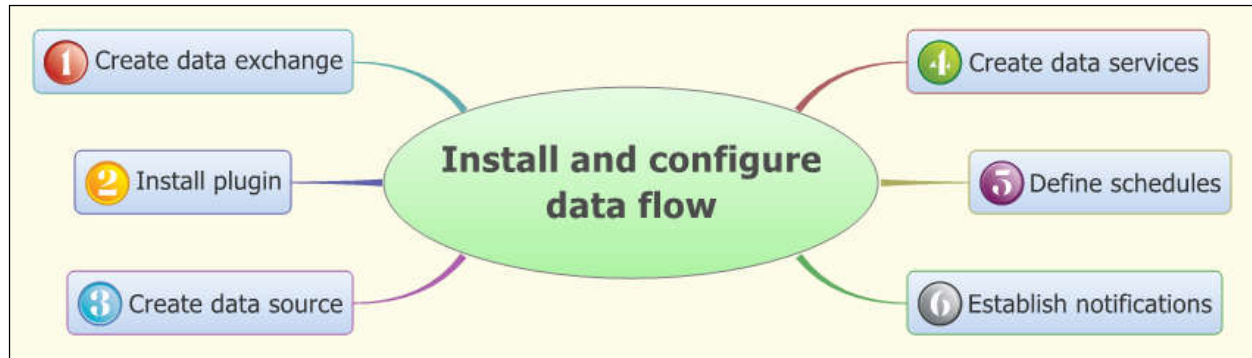


1. The first step is to create the staging database itself if one has not already been established to support another data exchange (typically named NODE_FLOW).
2. Once the staging database is created, a Database Definition Language (DDL) script included in the BEACHES plugin package can be executed 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:

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/data-exchange/beach-notification/>.

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:

5. Click the **Choose File** 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** 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.

The screenshot shows the 'Data Exchange Manager' interface with the 'Manage Exchange Service' tab selected. The form is for the 'BEACHES' exchange. It includes the following fields and options:

- Name:** BeachesExtractAndSubmission
- Implementer:** BeachesExtractAndSubmission (v2.7.3.1152)
- Type:** Task
- Active:** ☒ (Note: Making this service inactive will prevent it from being accessible using the Web Service interface.)
- Arguments:**
 - Execute Timeout (in seconds):** 900 (with a 'Use global value' checkbox)
 - Extract Stored Procedure Name:** etl_notif_beach (with a 'Use global value' checkbox)
- Data Sources:** Data Source (with a dropdown menu showing 'NODE_FLOW_WQX_AND_BEACHES')

At the bottom right, there are 'Cancel', 'Save', and 'Delete' buttons.

13. In the **Service Name** field, type “BeachesExtractAndSubmission”.
14. From the **Implementer** drop down box, select the value “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 exchange. 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:

Implementer:

Type:

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

Data Sources: **Data Source**

3. In the **Service Name** field, type “GetBeachesSubmissionStatus”.
4. From the **Implementer** drop down box, select the value “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 exchange. 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:

Implementer:

Type:

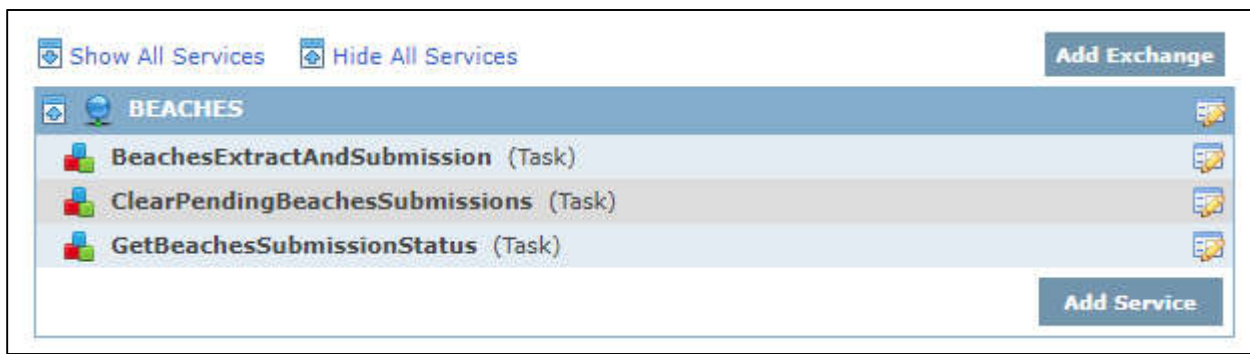
Active: ☒ Note: Making this service inactive will prevent it from being accessible using the Web Service interface.

Data Sources: Data Source

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:



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:

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: BeachesExtractAndSubmission ☒ Active

Exchange: BEACHES

Availability: Starts On: 2016-11-25 Ends On: 2026-06-30 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	Value	
AttributeEffectiveYear	2017	+ -
UseSubmissionHistoryTable	true	+ -

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: EPA NGN v2.0 (Prod) (Node v2.0)

Exchange: BEACHES

Operation:

Endpoint User: Node Runtime

Audit: Last modified by bill_rensmith@windsorsolutions.com on 12/6/2017 2:01:53 PM

Cancel Save Delete Save and Run Now

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, for example.
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 = set to current year
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 list of EPA endpoints](#) for a list of appropriate CDX endpoints for the BEACHES exchange.
11. In the **Exchange** field, type in “BEACHES”.
12. Leave **Operation** blank
13. Set the **Endpoint User** to “Node Runtime”. See the OpenNode2 Admin guide for more information on this setting.
14. 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:

Manage Schedules

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:
2016-11-29
Ends On:
2026-07-08
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 - 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 bill_rensmith@windsorsolutions.com on 12/6/2016 4:55:05 PM

Cancel
Save
Delete
Save and Run Now

- Type "GetBeachesSubmissionStatus" in the **Name** field.
- Enable the schedule by clicking the **Active** checkbox.
- Select "BEACHES" from the **Exchange** dropdown list.
- This is a manual schedule, and so the **Availability** and **Frequency** can be ignored. Although to "save and now", the frequency "once" should be selected.
- In the **Data Source** area, check the radio button labeled **Results of local service execution**.
- In the **Data Source** "from" dropdown box, select the value "GetBeachesSubmissionStatus."
- In the **Result Process** area, check the "None."
- 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

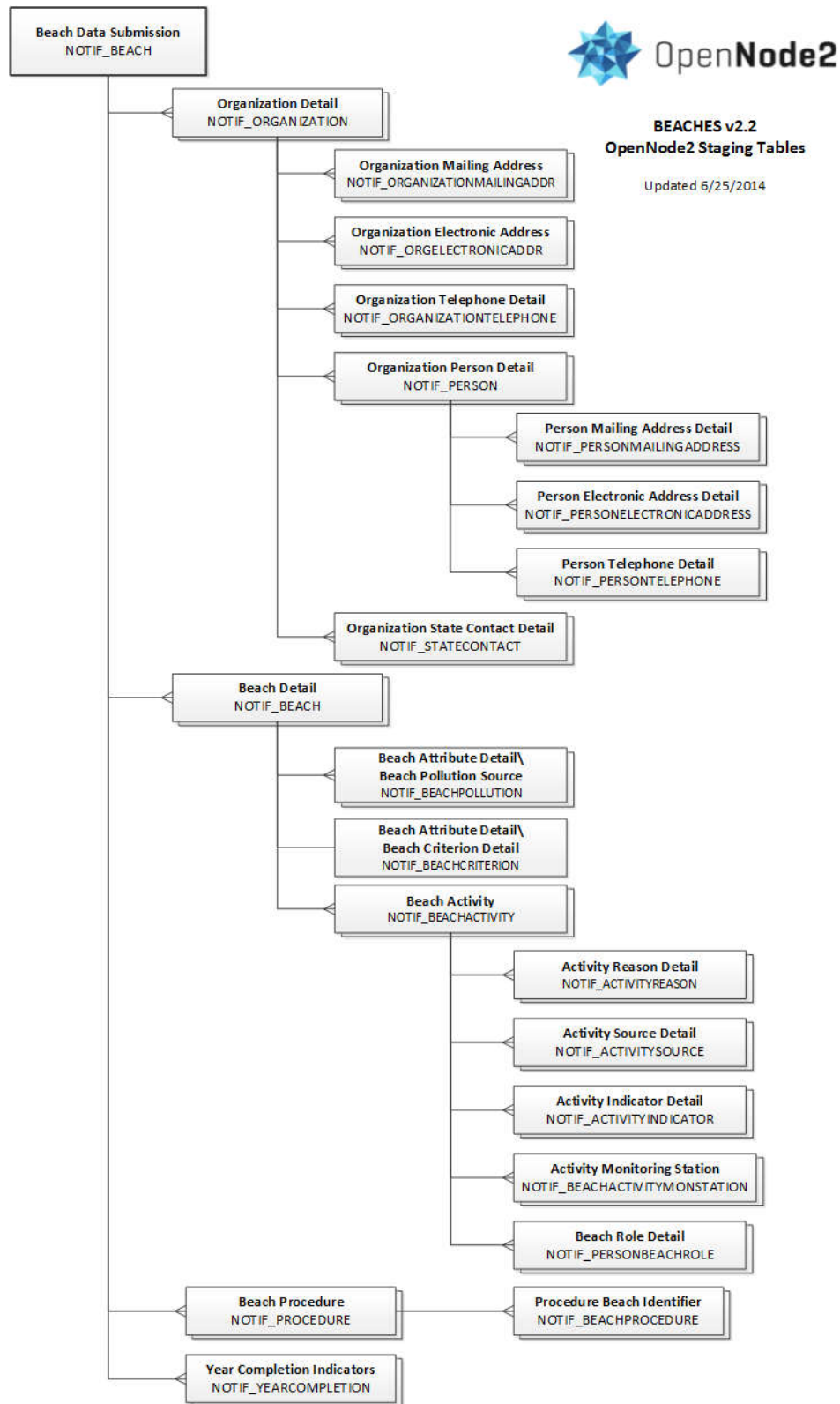
Contact the EPA CDX Node helpdesk and ask them to perform the following tasks:

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



Appendix B: Input Parameter table for GetNotificationDetailsByParameters

The GetNotificationDetailsByParameters service can be configured to expose query capabilities to external partners. The table below describes the available filters to retrieve specific notification data from OpenNode2.

Input Parameter	Required (Y/N)	Type	Wildcards (Y/N)	Notes
ActualStartDate	N	Date	N	Returns notifications that begin on or after the date supplied Example: “6-30-2015”
ActualStopDate	N	Date	N	Returns notifications that end on or before the date supplied Example: “10-1-2015”
BeachName	N	String	Y	Example: “Castle beach” Or “Newcastle beach” Or “castle”
BeachIdentifier	N	String	N	Example: MI159835
MonitoringStationIdentifier	N	String	N	Example: WQX123
BeachAccessType	N	String	N	Limited to enumeration values supported by Beach Notification schema Example: “PRV_PUB_ACC” Or “PUB_PUB_ACC”
ActivityIndicatorType	N	String	N	Limited to enumeration values supported by Beach Notification schema Example: “ECOLI” Or “FECAL”
ActiveAdvisoryIndicator	N	Boolean	N	If true, will return only the active advisories

				Example: “True” or “False”
--	--	--	--	-------------------------------