To Do:

Mass Delete, Reissuance does not result in creation of limits, add State implementation responsibilities section.

Environmental Council of States

OpenNode2 ICIS-NPDES Full Batch Data Flow Plugin

Design Specification

Version 1.0

Revision Date: January 4, 2012

WINDSOR SOLUTIONS, INC.
Environmental + Health Information Systems

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

1 Overview

This document describes the technical architecture and design of the ICIS-NPDES Full Batch Plugin and data flow for .NET OpenNode2.

This section to be completed.

2 Data Exchange Overview

The ICIS-NPDES full batch plugin operates in three overarching stages:

1. **Data Preparation Stage**
2. **Submission Stage**
3. **Result Processing Stage**

The **Data Preparation Stage** involves refreshing the staging database with the latest data from the state's NPDES information system. The plugin components will automatically determine what data from the staging database needs to get sent to ICIS-NPDES, alleviating the ETL logic from needing to perform this task.

The **Submission Stage** involves retrieving new, changed, or deleted data from the staging tables, converting the data to ICIS-NPDES XML, and transmitting the XML to EPA's Central Data Exchange (CDX). This stage is executed by a scheduled task within OpenNode2.

The **Result Processing Stage** involves retrieving ICIS-NPDES processing reports in XML from EPA's CDX system and parsing the accepted and rejected transactions into a tracking table. Accepted records are copied to a set of staging tables that are used to reflect the current data in ICIS and used to determine the data that needs to be sent in subsequent submissions. This stage is executed by a scheduled task within OpenNode2.

Each stage is executed in isolation and is only responsible for one aspect of the exchange. This design is intended to maintain a separation of concerns, allowing each component to focus only on a single task. By extension, each component should have little or no dependency on the specific tasks performed by the other stages.

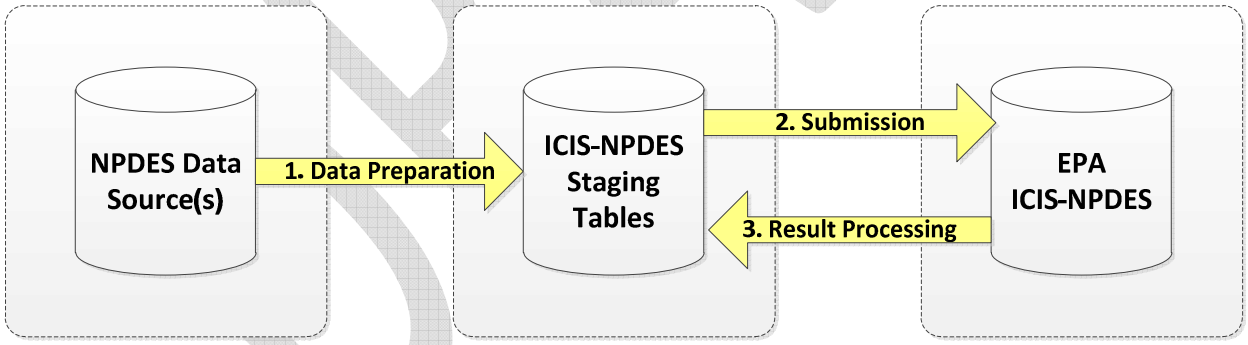


Figure 1 : Workflow Processing Stages

3 Staging Table Architecture

Data used in the exchange is stored in the ICIS-NPDES staging tables. There are four groupings of staging tables, each used for a different purpose:

1. Staging-Local
2. Staging-ICIS
3. Result Tracking
4. Transaction Tracking

More detailed descriptions of each data area are included in subsequent sections of this document.

3.1.1 Staging-Local

Staging-Local is used to store a reflection of the current ICIS-NPDES data in the state's own data systems. It is populated by a state-specific Extraction, Transformation, and Loading (ETL) routine on a regular interval.

The Staging-Local tables closely follow the structure of the ICIS-NPDES XML Schema. Staging-Local tables are stored in the "local" schema. The data in these tables is populated by a state-specific ETL routine that reads data from the source NPDES information management system(s), translates the data into ICIS-NPDES structures and formats (including use of proper lookup values and business rules), and inserts the transformed data into these tables.

The Staging-Local tables contain persistent data. These tables should always contain a full representation of the data that the state wishes to be represented in ICIS-NPDES for all data families. That is to say, data should not be removed from these tables when submission of a given data set is complete. The change detection process (used to determine what data to submit) is dependent upon having all historical data present in the Staging-Local tables.

3.1.2 Staging-ICIS

The Staging-ICIS tables are used to store all the records that have been accepted by EPA ICIS-NPDES. The table structures are identical to that of Staging-Local, which again, are a reflection of the ICIS-NPDES XML schema. Staging-Local tables are stored in the "ICIS" schema.

When an Accepted Transactions XML Report is processed by OpenNode2, accepted data are moved from Staging-Local to Staging-ICIS. Subsequent submissions are generated by comparing the contents of Staging-Local with Staging-ICIS to determine which transactions need to be generated and submitted.

Data in Staging-ICIS should never be manually manipulated. The data in this schema is maintained entirely by logic embedded in the plugin software and associated database routines. While data in this schema should not be altered, the state's ETL routines may wish to read data from this area as a means of helping to determine what data to add, change, or delete in Staging-Local.

3.1.3 Result Tracking Table (ICS_SUBM_RESULT)

The ICS_SUBM_RESULT table tracks data in both the Accepted Transactions and Rejected Transactions reports returned by ICIS-NPDES after a submission file has been processed. Data in this table is maintained by the logic within the OpenNode2 plugin responsible for retrieving and parsing processing results XML files. Data in this table should not be manually manipulated.

This table contains all the fields used across all submission types to track the natural keys associated with an accepted/rejected record. Depending on the submission type related to a processing result record, different fields will be populated.

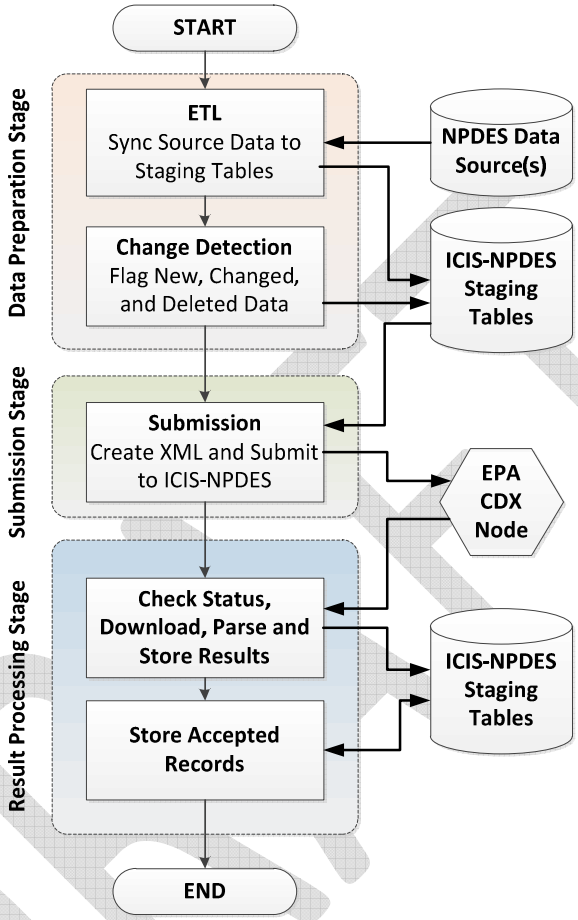
This table provides a single repository for all accepted and rejected transactions. The key fields related to a reject transactions are used by the built-in logic to flag records in Staging-Local for resubmittal. The key fields related to an accepted transaction trigger the built-in logic to copy accepted records from Staging-Local to Staging-ICIS.

3.1.4 Transaction Tracking Table (ICS_SUBM_TRACK)

The ICS_SUBM_TRACK table stores a record for each execution of the full data preparation, submission, and result processing lifecycle. Data in this table is maintained by the built-in data exchange logic and should not be manually manipulated. Detailed information about how this table is used to track lifecycle events is described in Section 4.1.2.

4 Workflow Processing

The following diagram decomposes the three overall workflow stages into sub-phases. The diagram below decomposes the three major stages into more detail.



A detailed description of each stage can be found in sections 4.2, 4.3, and 4.4, respectively.

4.1 Execution Management

4.1.1 Execution Sequencing Considerations

The processing stages must be executed in sequence. The prior sequence must complete before it can be re-executed. For example, the results of the previous submission must have completed processing before the data preparation stage can be executed again. This tight sequencing of events is important to retain a level of simplicity to the overall flow design and execution.

The first two stages (Data Preparation and Submission) can be executed as a single unit of work or they can be decoupled and triggered by independent events. The state agency will need to determine the best approach for their environment. The combined approach is recommended since it reduces the number of independent triggers that need to be managed and controlled. The separate approach is recommended during initial flow development because it allows the implementer to audit and adjust the staging routines and data as necessary before performing the submission of data to EPA.

To execute the first two stages as a unit, supply the name of the master ETL Stored Procedure as an argument in the plugin's SubmitICISData service. The plugin will run the ETL prior to sending XML data to EPA. If this argument is left blank, the ETL must be triggered by an independent process, such as a SQL Server Scheduled Job, chron job, or some other external trigger.

4.1.2 Execution State Management

The execution of the full lifecycle of data preparation, submission, and result processing is tightly controlled. The control apparatus is the ICS_SUBM_TRACK lifecycle tracking table. The plugin and ETL procedures read from and write to this table at various stages in the data flow lifecycle. The table contains the following fields and definitions

Field Name	Description
ICS_SUBM_TRACK_ID	A unique identifier associated with each lifecycle execution.
ETL_COMPL_DATE	The date and time when the ETL process completes as part of the Data Preparation Stage. This value is written as the last step of the ETL routine before control is passed to the Change Detection routine.
DET_CHG_COMPL_DATE	The date and time when the Change Detection process completes as part of the Data Preparation Stage. This value is written as the last step of the Change Detection routine.
SUBMIT_DATE	The date and time when the plugin submits the data for all ICIS-NPDES modules and records flagged during the Change Detection process. This value is written upon completion of the PerformICISSubmision service execution.
SUBMIT_STATUS	The Exchange Network status code reflecting the current status of the submission. Status will be either Pending, Completed or Failed. This value is written upon completion of the PerformICISSubmision service execution or upon execution of a GetStatus command within the GetStatusAndProcessReports service.
SUBMIT_STATUS_DATE	The date and time the status of a given submission was last verified from EPA CDX. This value is written upon completion of the PerformICISSubmision service execution or upon execution of a GetStatus command within the GetStatusAndProcessReports service.
RESPONSE_PARSE_DATE	The date and time the Accepted and Rejected Transactions reports were parsed and loaded into the ICS_SUBM_RESULT table. This value is written upon completion of the GetStatusAndProcessReports service

The following section describes the management of a single record in the ICS_SUBM_TRACK table through the execution of a single end-to-end data preparation, submission, and response processing lifecycle. In each step below, updated values are shown in bold, red text.

Step 1: ETL Execution. Upon successful completion of the state-specific ETL routine, a new record is created in ICS_SUBM_TRACK listing the completion date of the ETL Event:

ICS_ SUBM_ TRACK_ID	ETL_ COMPL_ DATE	DET_ CHG_ COMPL_ DATE	SUBMIT_ DATE	SUBMIT_ TRANS_ID	SUBMIT_ STATUS	SUBMIT_ STATUS_ DATE	RESPONSE_ PARSE_ DATE
ABC123	**2012-02-01 10:00:00**						

Step 2: Change Detection Execution. The change detection procedure updates transaction codes for each of the modules/records to be added, updated, or deleted from ICIS. Upon completion, the change detection procedure updates the row to include the time of completion.

ICS_ SUBM_ TRACK_ID	ETL_ COMPL_ DATE	DET_ CHG_ COMPL_ DATE	SUBMIT_ DATE	SUBMIT_ TRANS_ID	SUBMIT_ STATUS	SUBMIT_ STATUS_ DATE	RESPONSE_ PARSE_ DATE
ABC123	2012-02-01 10:00:00	**2012-02-01 10:13:45**					

Step 3: Submission of data to ICIS-NPDES. Upon completion of the submission process (performed by the PerformICISSubmission service), the plugin updates the record with the Submission date, transaction ID, and transaction status. The status will likely be pending upon initial completion.

ICS_ SUBM_ TRACK_ID	ETL_ COMPL_ DATE	DET_ CHG_ COMPL_ DATE	SUBMIT_ DATE	SUBMIT_ TRANS_ID	SUBMIT_ STATUS	SUBMIT_ STATUS_ DATE	RESPONSE_ PARSE_ DATE
ABC123	2012-02-01 10:00:00	2012-02-01 10:13:45	**2012-02-01 10:13:45**	**2AKSJ-19S9-S9K2-SL91K**	**PENDING**	**2012-02-01 10:14:12**	

Step 4: Get Submission Status. The GetStatusAndProcessReports service will check the status of any pending submissions against CDX. If the status has changed, the table is updated with the latest status value.

ICS_ SUBM_ TRACK_ID	ETL_ COMPL_ DATE	DET_ CHG_ COMPL_ DATE	SUBMIT_ DATE	SUBMIT_ TRANS_ID	SUBMIT_ STATUS	SUBMIT_ STATUS_ DATE	RESPONSE_ PARSE_ DATE
ABC123	2012-02-01 10:00:00	2012-02-01 10:13:45	2012-02-01 10:13:45	2AKSJ-19S9-S9K2-SL91K	**COMPLETED**	**2012-02-01 13:09:34**	

Step 5: Parse Results. The GetStatusAndProcessReports service updates the table with the date and time when it completes parsing the Accepted and Rejected Transactions reports into the ICS_SUBM_RESULT table.

ICS_ SUBM_ TRACK_ID	ETL_ COMPL_ DATE	DET_ CHG_ COMPL_ DATE	SUBMIT_ DATE	SUBMIT_ TRANS_ID	SUBMIT_ STATUS	SUBMIT_ STATUS_ DATE	RESPONSE_ PARSE_ DATE
ABC123	2012-02-01 10:00:00	2012-02-01 10:13:45	2012-02-01 10:13:45	2AKSJ-19S9-S9K2-SL91K	COMPLETED	2012-02-01 13:09:34	**2012-02-01 13:10:31**

At this stage, the execution lifecycle is complete for a single set of changes.

4.2 Data Preparation Stage

The data preparation stage is the first stage in the overall workflow. This stage is composed of two discrete sub-stages;

1. a state-specific ETL process, and
2. a built-in change detection process

4.2.1 Populating Staging-Local Tables

As stated previously, Staging-Local is used to store a reflection of the current ICIS-NPDES data in the state's own data systems. It is populated by a state-specific Extraction, Transformation, and Loading (ETL) routine on a regular interval.

States will need to build the ETL routines to populate the data in the Staging-Local tables. The architecture and design of the state's ETL routine is flexible. States may choose from a variety of approaches, so long as the end result is the same. Upon completion of the ETL process, the data Staging-Local must represent a reflection of the current NPDES data in the state system for all modules being flowed. States may wish to completely rebuild the data each time the ETL runs, or alternatively, incrementally insert, update, or delete records to bring the data up to date.

One of the major advantages of the dual-schema design is it frees the state from needing to be concerned with which data needs to be sent to ICIS since this is entirely managed within the built-in logic. The only concern is that the data in Staging-Local is current.

4.2.1.1 ICS_PAYLOAD Table

The root table is ICS_PAYLOAD. This should contain a record for each of the submission types to be transmitted to ICIS-NPDES. For example, if a state does not wish to send enforcement data to ICIS-NPDES, then do not create a record for these payload types.

The submission type is stored in the OPERATION field of this table[1]. These records trigger the XML generation process to include a given data family in the submission. The allowable submission types are listed in the XML Schema User's Guide and closely follows the name of the XML element name of the given submission type. The table below lists each of the allowable submission types:

Category	Submission Type	Operation Name	Root Staging Table Name
Category	Basic Permit	BasicPermitSubmission	ICS_BASIC_PRMT
Permit	Biosolids Permit	BiosolidsPermitSubmission	ICS_BS_PRMT
Permit	CAFO Permit	CAFOPermitSubmission	ICS_CAFO_PRMT
Permit	CSO Permit	SCOPermitSubmission	ICS_CSO_PRMT
Permit	General Permit	GeneralPermitSubmission	ICS_GNRL_PRMT
Permit	Master General Permit	MasterGeneralPermitSubmission	ICS_MASTER_GNRL_PRMT
Permit	Permit Reissuance	PermitReissuanceSubmission	ICS_PRMT_REISSU
Permit	Permit Termination	PermitTerminationSubmission	ICS_PRMT_TERM
Permit	Permit Tracking Event	PermitTrackingEventSubmission	ICS_PRMT_TRACK_EVT
Permit	POTW Permit	POTWPermitSubmission	ICS_POTW_PRMT
Permit	Pretreatment Permit	PretreatmentPermitSubmission	ICS_PRETR_PRMT
Permit	SWConstruction Permit	SWConstructionPermitSubmission	ICS_SW_CNST_PRMT
Permit	SWIndustrial Permit	SWIndustrialPermitSubmission	ICS_SW_INDST_PRMT
Permit	SW MS4 Large Permit	SWMS4LargePermitSubmission	ICS_SWMS_4_LARGE_PRMT

[1] See Section 3.5.2 Payload Block in the ICIS XML Schema User's Guide for more information.

Category	Submission Type	Operation Name	Root Staging Table Name
Permit	SW MS4 Small Permit	SWMS4SmallPermitSubmission	ICS_SWMS_4_SMALL_PRMT
Permit	Unpermitted Facility	UnpermittedFacilitySubmission	ICS_UNPRMT_FAC
Permit	Historical Permit Schedule Events	HistoricalPermitScheduleEventsSubmission	ICS_HIST_PRMT_SCHD_EVTS
Permit	Narrative Condition Schedule	NarrativeConditionScheduleSubmission	ICS_NARR_COND_SCHD
DMR	Permitted Feature	PermittedFeatureSubmission	ICS_PRMT_FEATR
DMR	Limit Set	LimitSetSubmission	ICS_LMT_SET
DMR	Limits	LimitsSubmission	ICS_LMTS
DMR	Parameter Limits	ParameterLimitsSubmission	ICS_PARAM_LMTS
DMR	Discharge Monitoring Report	DischargeMonitoringReportSubmission	ICS_DSCH_MON_REP
DMR	Effluent Trade Partner	EffluentTradePartnerSubmission	ICS_EFFLU_TRADE_PRTNER
Inspection	Compliance Monitoring	ComplianceMonitoringSubmission	ICS_CMPL_MON
Inspection	Compliance Monitoring Linkage	ComplianceMonitoringLinkage	ICS_CMPL_MON_LNK
Enforcement	Formal Enforcement Action	FormalEnforcementActionSubmission	ICS_FRML_ENFRC_ACTN
Enforcement	Informal Enforcement Action	InformalEnforcementActionSubmission	ICS_INFRML_ENFRC_ACTN
Enforcement	Final Order Violation Linkage	FinalOrderViolationLinkageSubmission	ICS_FINAL_ORDER_VIOL_LNK
Enforcement	Compliance Schedule	ComplianceScheduleSubmission	ICS_CMPL_SCHD
Enforcement	Enforcement Action Milestone	EnforcementActionMilestoneSubmission	ICS_ENFRC_ACTN_MILESTONE
Violation	Enforcement Action Violation Linkage	EnforcementActionViolationLinkageSubmission	ICS_ENFRC_ACTN_VIOL_LNK
Violation	DMR Violation	DMRViolationSubmission	ICS_DMR_VIOL
Violation	Single Event Violation	SingleEventViolationSubmission	ICS_SNGL_EVT_VIOL
Violation	Schedule Event Violation	ScheduleEventViolationSubmission	ICS_SCHD_EVT_VIOL
Report	CSO Event Report	CSOEventReportSubmission	ICS_CSO_EVT_REP
Report	SW Event Report	SWEventReportSubmission	ICS_SW_EVT_REP
Report	DMR Program Report Linkage	DMRProgramReportLinkageSubmission	ICS_DMR_PROG_REP_LNK
Report	CAFO Annual Report	CAFOAnnualReportSubmission	ICS_CAFO_ANNUL_REP
Report	Local Limits Program Report	LocalLimitsProgramReportSubmission	ICS_LOC_LMTS_PROG_REP
Report	Pretreatment Performance Summary	PretreatmentPerformanceSummarySubmission	ICS_PRETR_PERF_SUMM
Report	Biosolids Program Report	BiosolidsProgramReportSubmission	ICS_BS_PROG_REP
Report	SSO Annual Report	SSOAnnualReportSubmission	ICS_SSO_ANNUL_REP
Report	SSO Event Report	SSOEventReportSubmission	ICS_SSO_EVT_REP
Report	SSO Monthly Event Report	SSOMonthlyEventReportSubmission	ICS_SSO_MONTHLY_EVT_REP
Report	SW MS4 Program Report	SWMS4ProgramReportSubmission	ICS_SWMS_4_PROG_REP

4.2.1.2 Unique Business Keys

The 46 root submission tables in Staging-Local and Staging-ICIS contain unique constraints on each of the business key fields. For example, the Basic Permit table (ICS_BASIC_PRMT) has a unique constraint on the Permit Number (PRMT_IDENT). This ensures that each permit only ever exists once in the staging table. It preserves data integrity in the staging environment and provides a critical foundation for the ETL design and the automated change detection processes.

4.2.1.3 Source System Identifier Tracking

As a convenience, each of the 46 submission root tables contains a field named SRC_SYSTM_IDENT to track the unique identifier for the record from the state's source data system. The use of this field is optional. It is ignored by the plugin XML generation code. State ETL routines may wish to join on this field as part of the ETL process to determine what data needs to be changed in the Staging-Local environment.

4.2.1.4 Asterisks

The ICIS-NPDES Batch flow at EPA supports the use of asterisks in field data to represent fields that should be "blanked out" in ICIS-NPDES. EPA only supports asterisks in Change transactions.

The plugin staging tables use strong data typing wherever possible, such as Date and Numeric fields that eliminates the ability to use asterisks in some cases. This decision was made for the staging environment because strong typing greatly improves data integrity at the expense of not using asterisks. Also, Change transactions are only created by the change detection routine in very few cases, further reducing the potential impact of not supporting asterisks.

4.2.1.5 ETL Halt Trigger

The design of the ICIS-NPDES plugin components require that prior submissions complete (including the processing of Accepted and Rejected Transaction Reports) before a subsequent exchange can be executed. For this reason, it is necessary for the ETL routine to first ensure that the prior execution has completed before executing a new refresh of Staging-Local data.

To detect whether the prior submission has completed, the ETL should check the Transaction Tracking audit table. If the prior transaction has not completed, it must abort the ETL process without manipulating and data in the Staging-Local tables.

4.2.2 Change Detection (Automated)

Bundled with the data exchange software is a database procedure that is responsible for determining which data to send to ICIS-NPDES based on what has successfully been sent previously. After the state-built ETL process finishes refreshing the data in Staging-Local the ETL should execute the pre-built change detection procedure.

4.2.2.1 Flagging Records for Submittal to ICIS-NPDES

Change detection is performed by comparing the data in Staging-Local with the data in Staging-ICIS based on the business key for each module and record. The generalized logic for the change detection process is as follows:

- **New:** If a record with a unique business key is present in Staging-Local that does not contain a matching record in Staging-ICIS, set the Transaction Code to "N".
- **Change:** If a record is present in Staging-Local that contains a matching record in Staging-ICIS with the same business key but different data in any other field in the table or child table(s), set the Transaction Code to "C" or "R". Transaction Code "C" will be used in the Basic Permit and Limits submission types. All other submission types will use Transaction Code "R".
- **Delete:** If a record is present in Staging-ICIS that does not have a matching business key in Staging-ICIS, the change detection procedure will insert a record in Staging-Local with Transaction Code "D" containing only the business key for the record.

Records are flagged for submittal using the TRANSACTION_CODE field in each of the 46 root submission tables. The change detection routine begins by setting all the TRANSACTION_CODE fields to null and then setting this field to the appropriate Transaction Code (N, C, R, etc.) as changes are identified in each module and record using the generalized logic above. The presence of a Transaction Code in this field triggers the plugin submission service to include the record. Records with Null Transaction Codes are not sent.

The Mass Delete transaction (X) is not used since it is assumed that the ETL process will remove the child data from subsequent child modules, thus triggering the change detection procedure to generate Delete (D) transactions for each subsequent child record, negating the need for Mass Delete.

4.2.2.2 Considerations for Permit Reissuance

The most consequential data submission in the ICIS-NPDES flow is the Permit Reissuance submission. When a permit is reissued, a new copy of the permit is created in ICIS-NPDES and its existing data (and any active limit sets) are carried forward to the new version. The Reissuance submission contains only four required data elements; Permit ID, Issued, Effective and Expiration Date.

The change detection procedure must only set the Transaction Codes for data relating to permits that either do not exist in the ICS_PRMT_REISS table or whose reissuances have been accepted by ICIS-NPDES. This requires a match on all four fields between ICS_PRMT_REISS and ICS_BASIC_PRMT.

This is to avoid a situation where the plugin begins submitting data for a new version of a permit that has not been successfully reissued in ICIS-NPDES, thus improperly overwriting the data for the previous version.

4.2.2.3 ETL Completion Logging

The final step of the change detection procedure is to record the completion of the ETL process. The presence of this record is used as a trigger by the submission process to proceed. If this record is not present, submission of data will not occur. This record is inserted after the change detection procedure completes executing.

4.2.3 Pre-submission Validation

The EPA ICIS-NPDES batch processing logic contains several hundred business rules as documented in the EPA ICIS Batch Technical Specifications[2]. A stand-alone data validation script (ICS_VALIDATE) is included in the plugin package to allow states to perform some high-level validation of data in the staging environment before executing the submission. The validation script is not comprehensive, but it does offer the ability to capture some data issues before performing a submission. For example, the validation script cannot check for invalid ICIS lookup values or perform data integrity checks across submission types (such as sending a permit narrative condition record for a permit that does not exist in ICIS).

The validation script is provided as a manual QA tool. The validation script is not executed as part of the regular processing workflow. It is assumed that the validation script will be especially helpful for validating the ETL processes during development and implementation of the exchange.

[2] The ICIS Batch Technical Specifications are available for download from EPA's extranet portal. Please contact EPA for instructions on accessing these documents.

4.3 Submission Stage

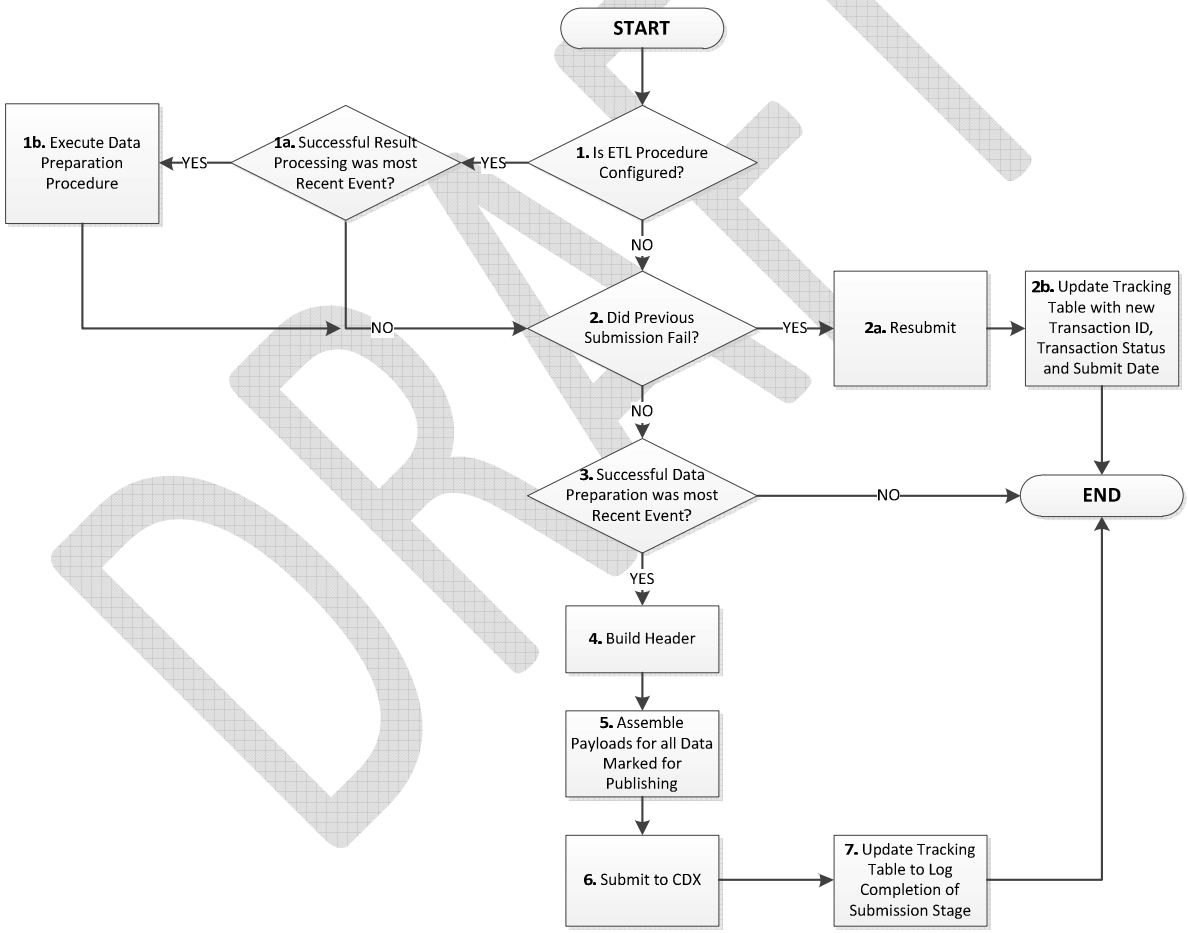
The OpenNode2 ICIS-NPDES Full Batch plugin is responsible for executing the submission of data to EPA CDX. The submission of data is performed by a single service within the ICIS-NPDES Full Batch plugin. This service is defined below.

4.3.1 PerformICISSubmission Service

The following sections describe with workflow and service configuration parameters for the PerformICISSubmission service.

4.3.1.1 Workflow

The ICIS-NPDES Full Batch OpenNode2 plugin contains a service named PerformICISSubmision responsible for sending data to ICIS-NPDES from the staging environment. The workflow diagram below illustrates the steps performed by the service.



Each step in the diagram is described in more detail below:

1. If the plugin has been configured to execute the ETL (determined by the presence of a stored procedure name in the service parameter), then execute the ETL stored procedure.
    a. The service checks the ICS_SUBM_TRACK table to see if the most recent completed task was the Result Processing stage was the most recently completed event. If not, proceed directly to Step 2.

   b. Execute the Data Preparation Stage as described in Section 4.2.
2. The service checks the ICS_SUBM_TRACK table to see if the most recent event was a failed submission.
   a. If the transaction failed, rebuild and resubmit the submission file based on the records containing Transaction Codes in the Staging-Local database. The detailed steps are also described in 3 and 4 below.
   b. Update the record in ICS_SUBM_TRACK with a Submit Status and a Submit Date equal to the current date/time.
3. The service checks the ICS_SUBM_TRACK table to see if the most recent completed task was the Data Preparation Stage. If not, exit.
4. Assemble the XML submission Header block using the parameters provided in the service configuration to populate the header fields (Author, Organization, ContactInfo, etc.)
5. For each record in ICS_PAYLOAD, assemble the data to submit for each of the 46 child submission types where records contain Transaction Codes. The returned records are inserted into the XML payload blocks beneath the parent Document element. Validate the file against the ICIS-NPDES XML Schema. Upon success, compress into a zip file in preparation for submission.
6. Authenticate to the Node Partner configured in the service and submit the XML document to the partner.
7. Update ICIS_SUBM_TRACK table, setting the Submit Status to the status returned by CDX (Pending) and Submit Date equal to the current date/time.

4.3.1.2 Service Parameters

The PerformICISSubmission plugin service contains the following configuration parameters:

Parameter Name	Description
ETL Procedure Name	The name of the ETL stored procedure. If blank, assume it is executed independently.
Author	The text to insert into the Header element's Author tag
Organization	The text to insert into the Header element's Organization tag
Contact Info	The text to insert into the Header element's ContactInfo tag
Submission Partner Name	The name of the Network Partner configured in OpenNode2 for the CDX endpoint that the submission data will be sent to.
Data Source	The name of the Data Source configured in OpenNode2 representing a connection to the Staging-Local tables for the ICIS-NPDES exchange.

4.4 Result Processing Stage

The OpenNode2 ICIS-NPDES Full Batch plugin is responsible for checking the status of prior submissions and processing Accepted/Rejected Transaction Reports. This task is performed by a single service within the ICIS-NPDES Full Batch plugin. This service is defined below.

4.4.1 Submission Result Table

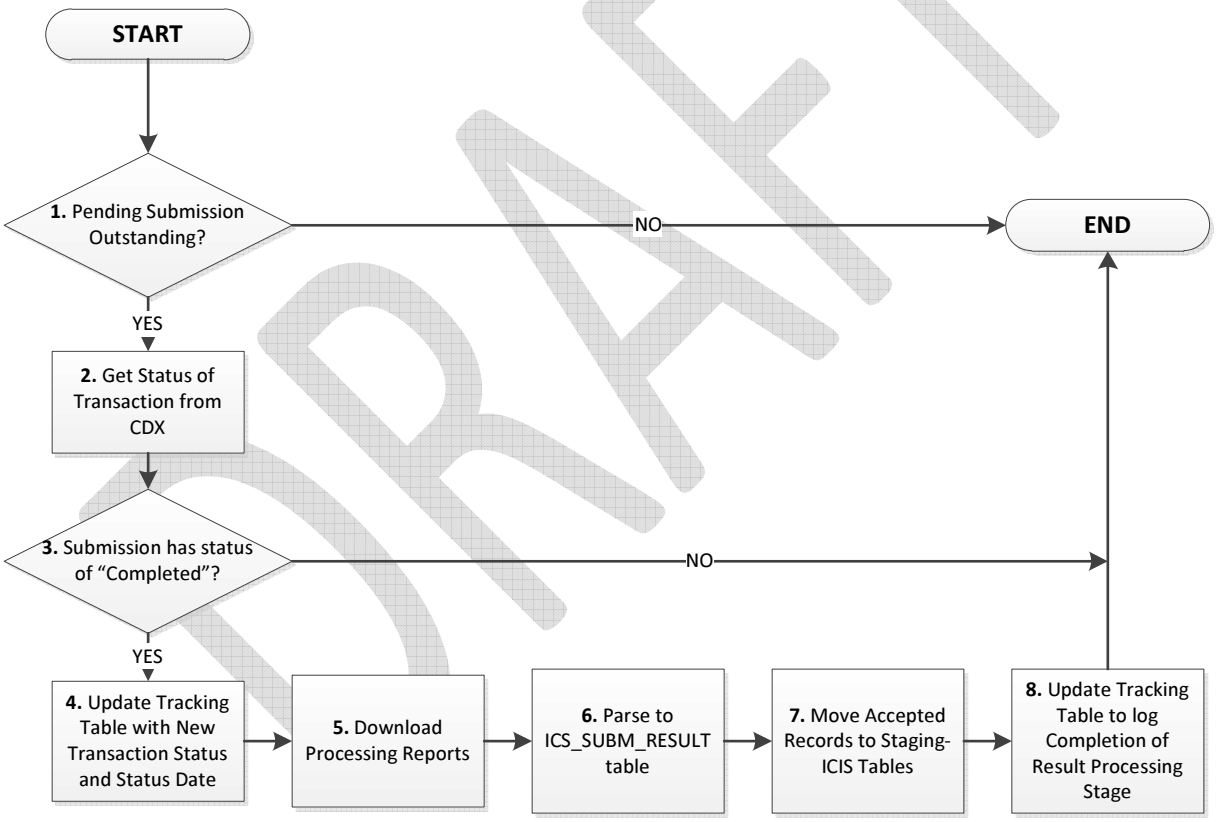
The ICIS-NPDES plugin staging table schema contains a table, ICS_SUBM_RESULT, used to store the accepted and rejected transactions from the most recent submission.

4.4.2 GetStatusAndProcessReports Service

The following sections describe with workflow and service configuration parameters for the PerformICISSubmission service.

4.4.2.1 Workflow

The ICIS-NPDES Full Batch OpenNode2 plugin contains a service named GetStatusAndProcessReports responsible for checking the status of previous submissions and, if processing is completed, downloading and parsing the Accepted and Rejected Transactions reports and parsing the results into a result tracking table. The workflow diagram below illustrates the steps performed by the service.



Each step in the diagram is described in more detail below:

1. The service checks the ICS_SUBM_TRACK table to see if the most recent event was a submission to ICIS-NPDES that has a status of Pending. If not, exit.
2. Retrieve the status of the most recent submission to ICIS-NPDES from CDX.
3. Check to see if the transaction has completed. If not, exit.
4. Update ICIS_SUBM_TRACK table, setting the Submit Status to the status returned by CDX (Completed) and Submit Date equal to the current date/time.

5. Download the processing report from CDX. The file name is <TransactionID>_<SubmittingParty>_<SubmissionDate>_Response.zip. Within this file are three files: Accepted_Response, Rejected_Response, and Summary_Response.
6. Parse the Accepted_Response and Rejected_Response records into the ICS_SUBM_RESULT table. Before inserting, purge the contents of the ICS_SUBM_RESULT table.
7. Execute the SQL procedure that migrates accepted transactions from Staging-Local to Staging-ICIS.
8. Update ICIS_SUBM_TRACK table, setting the Result Parse Date/Time equal to the current date/time.

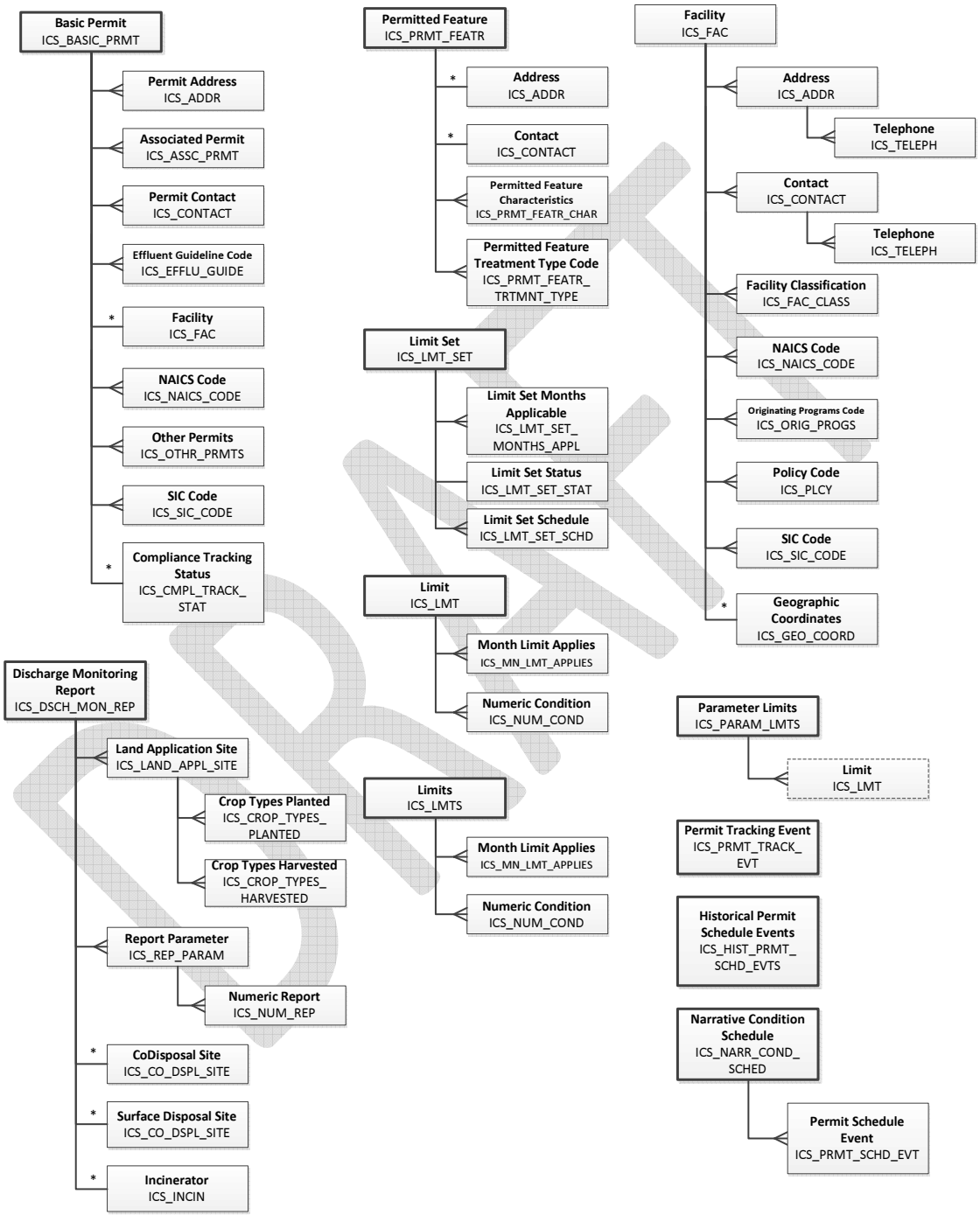
4.4.2.2 Service Parameters

The GetStatusAndProcessReports plugin service contains the following configuration parameters:

Parameter Name	**Description**
Submission Partner Name	The name of the Network Partner configured in OpenNode2 for the CDX endpoint that the submission data will be sent to.
Data Source	The name of the Data Source configured in OpenNode2 representing a connection to the Staging-Local tables for the ICIS-NPDES exchange.

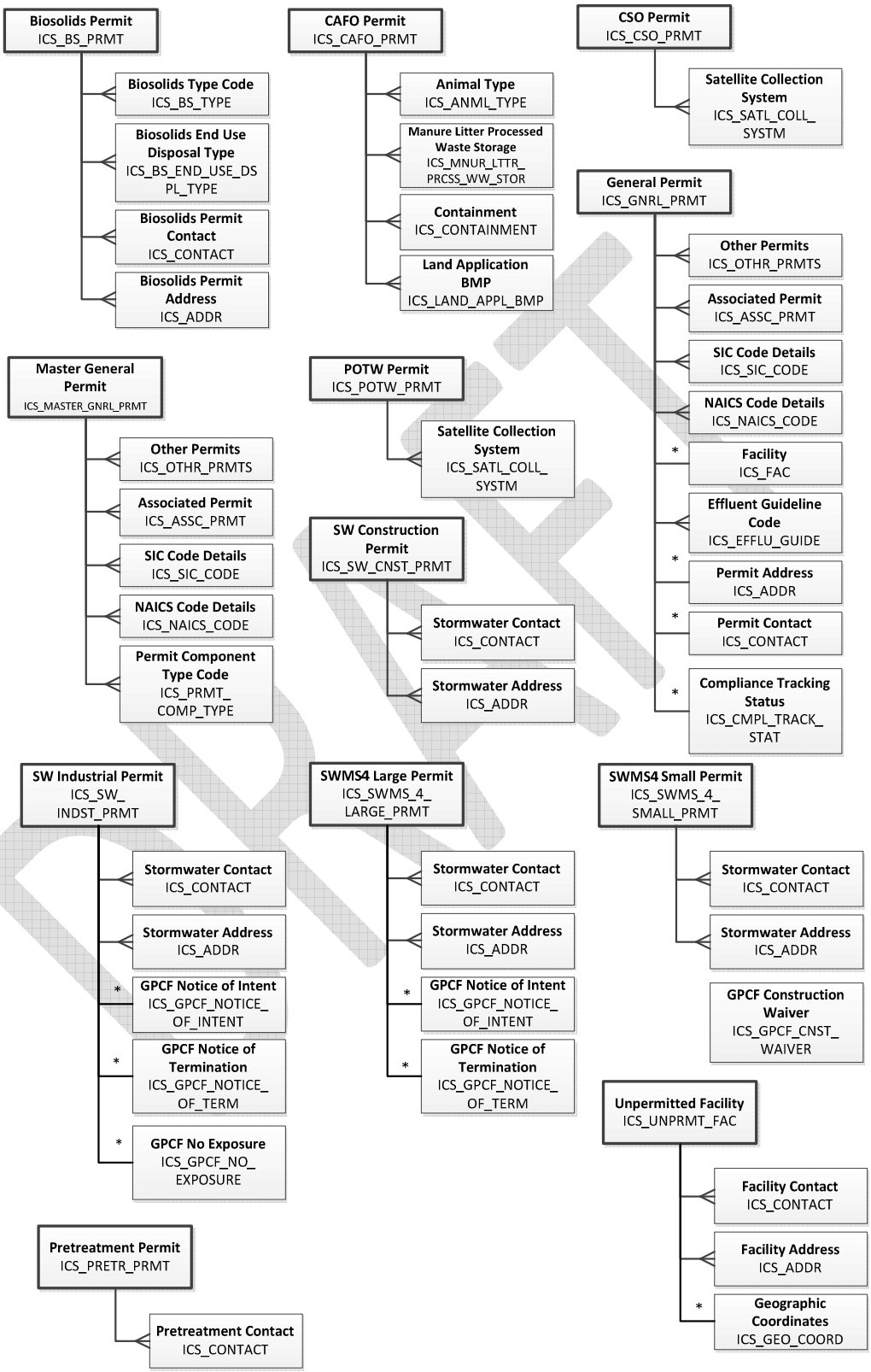
Appendix A: Table/Module Diagrams

Permit, Limit, DMR Module

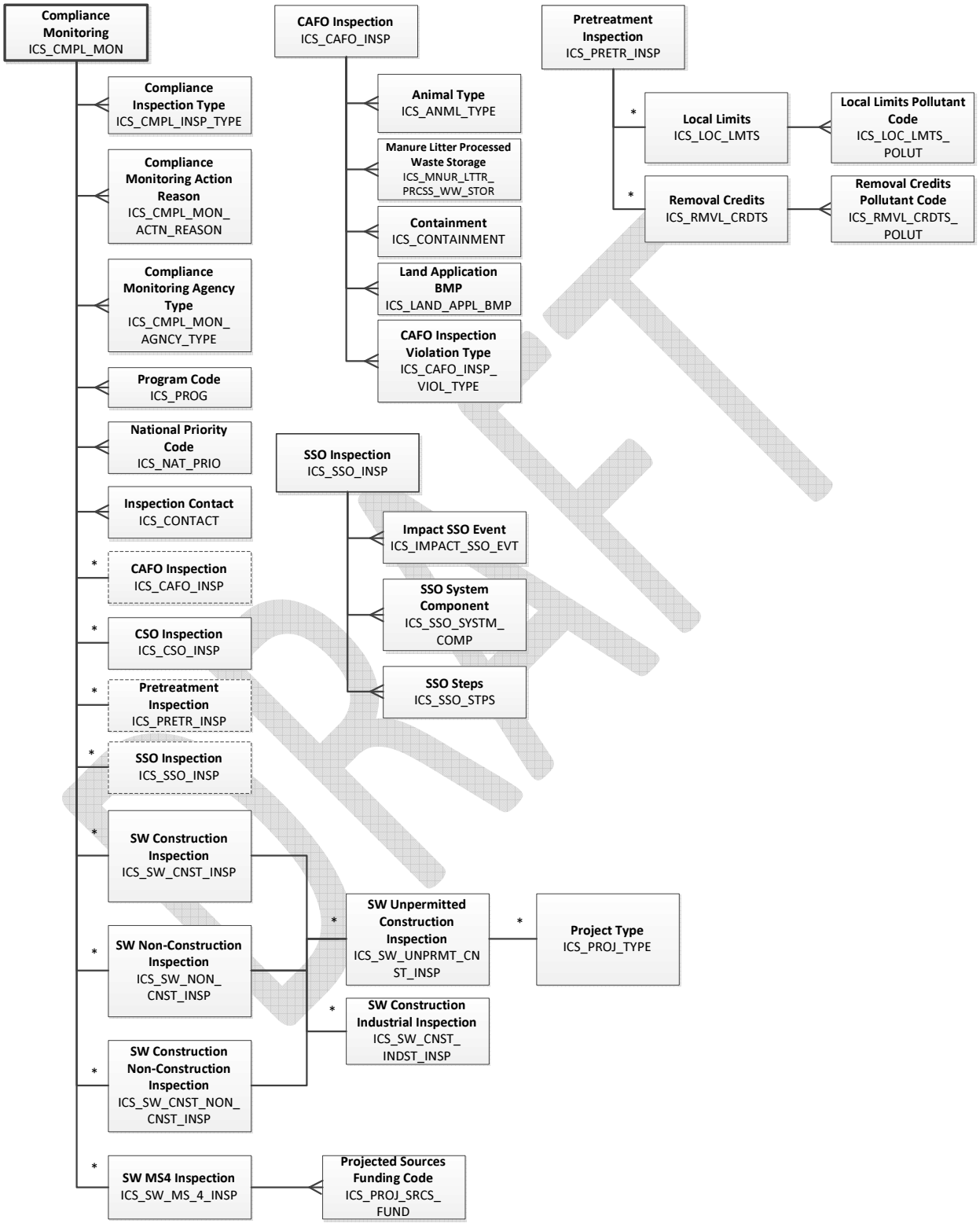


* XML Allows only one child, but data model supports many. Only populate one child record.

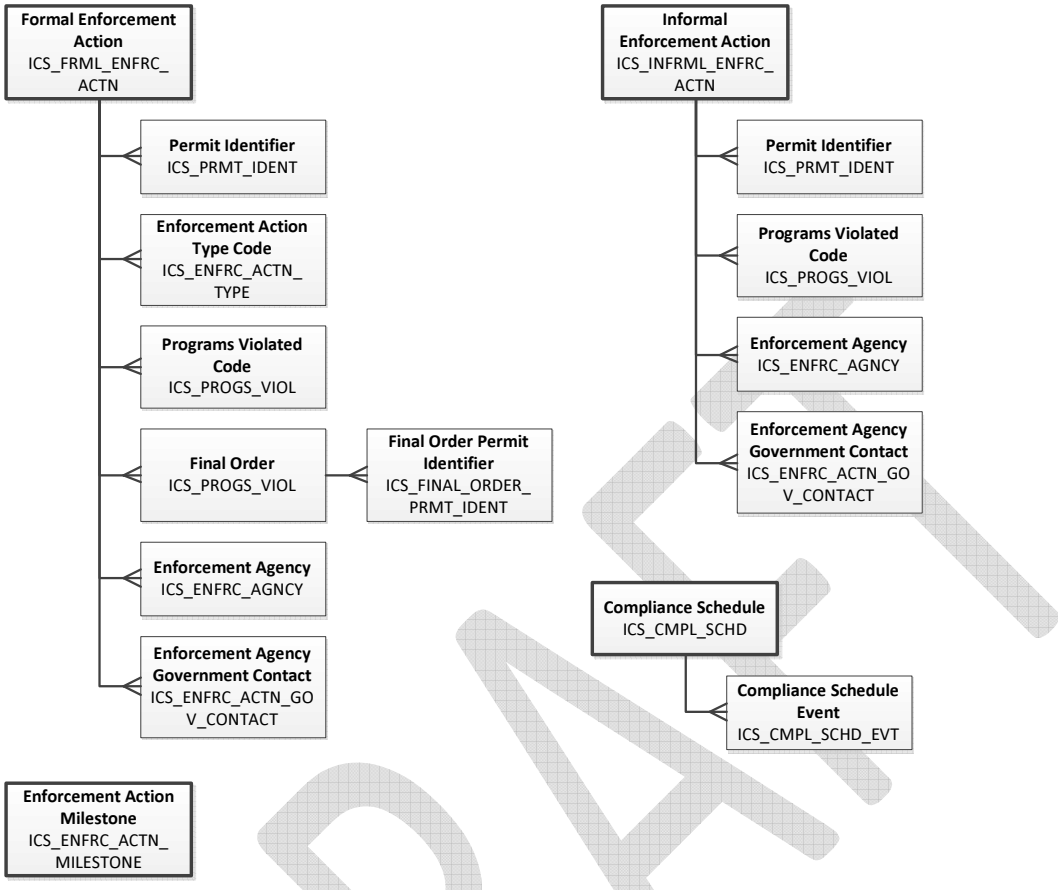
Permit Components



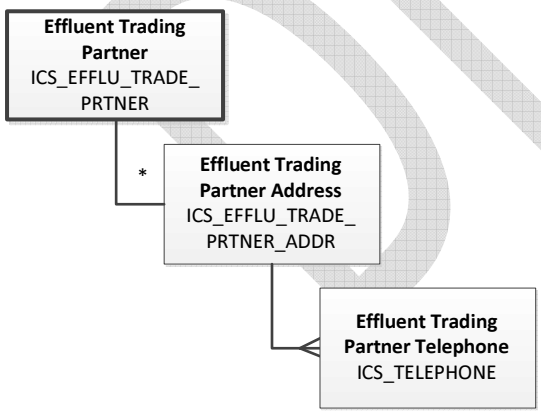
Inspections



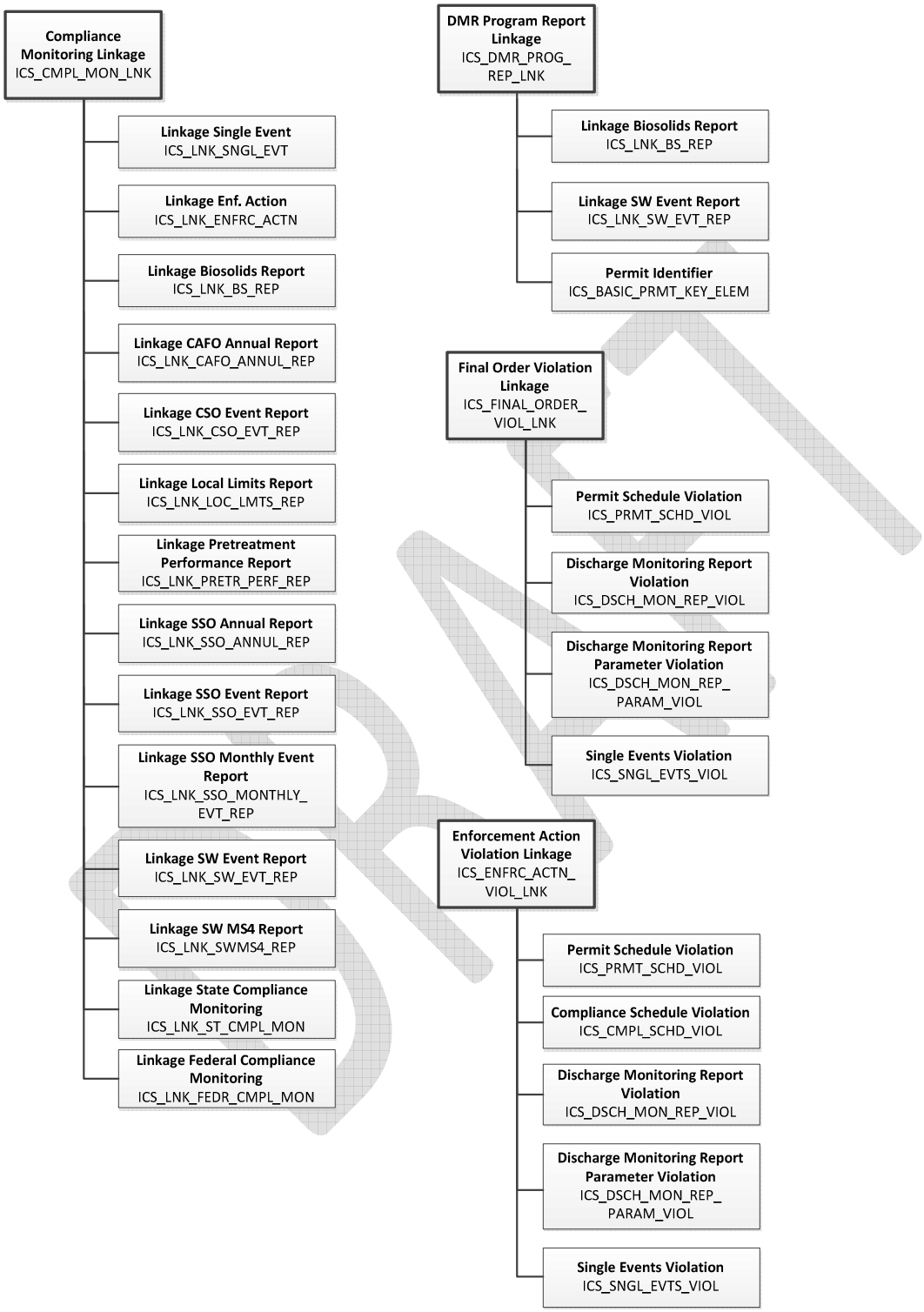
Enforcement



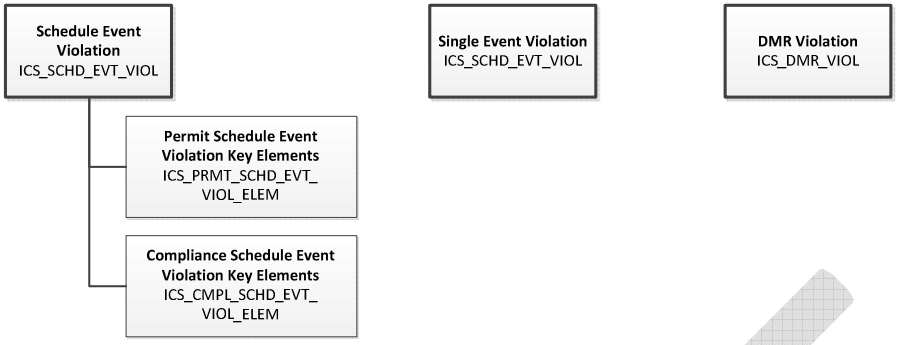
Effluent Trade Partner



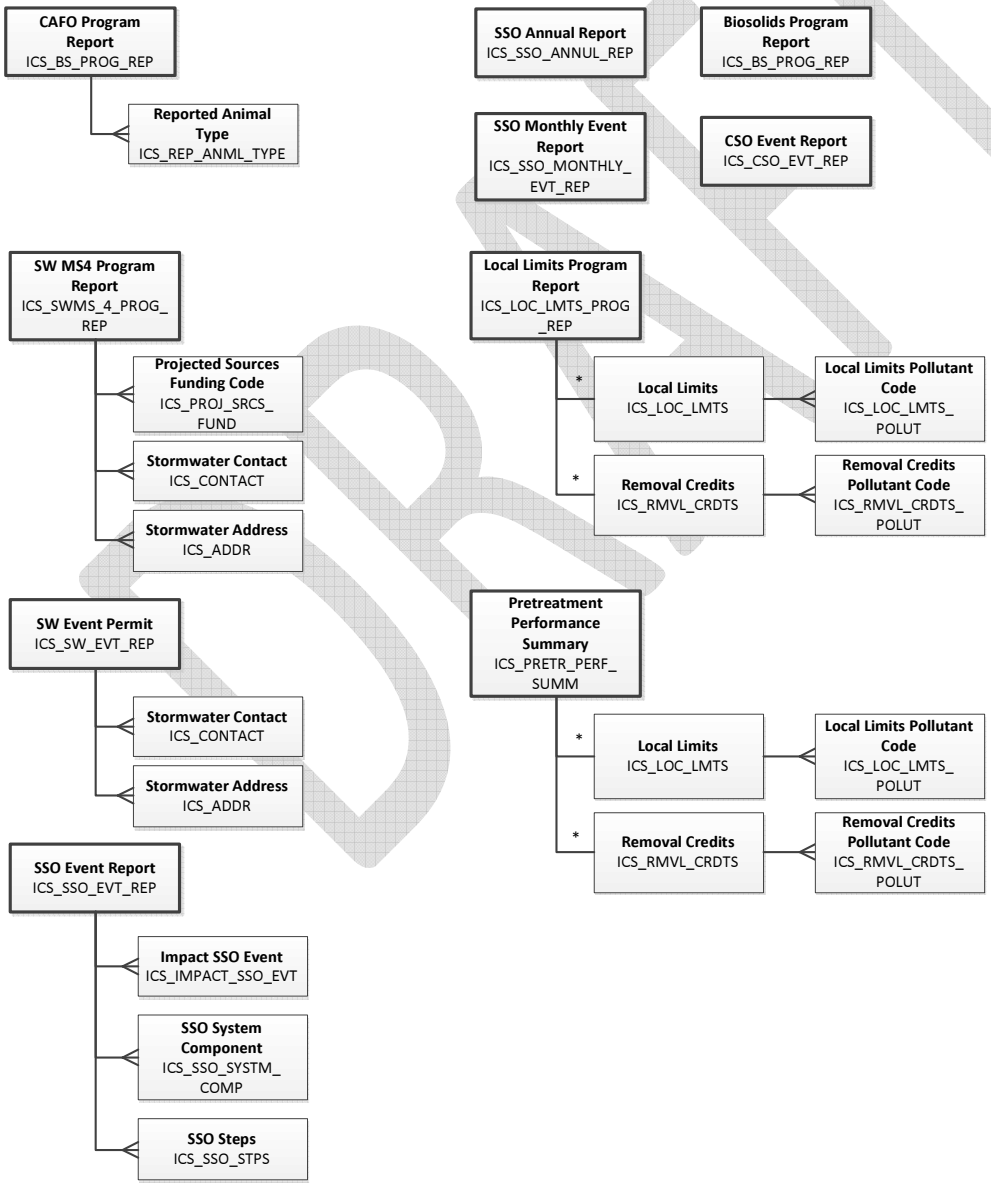
Linkages


Compliance Monitoring Linkage
ICS_CMPL_MON_LNK
Linkage Single Event
ICS_LNK_SNGL_EVT
Linkage Enf. Action
ICS_LNK_ENFRC_ACTN
Linkage Biosolids Report
ICS_LNK_BS_REP
Linkage CAFO Annual Report
ICS_LNK_CAFO_ANNUL_REP
Linkage CSO Event Report
ICS_LNK_CSO_EVT_REP
Linkage Local Limits Report
ICS_LNK_LOC_LMTS_REP
Linkage Pretreatment Performance Report
ICS_LNK_PRETR_PERF_REP
Linkage SSO Annual Report
ICS_LNK_SSO_ANNUL_REP
Linkage SSO Event Report
ICS_LNK_SSO_EVT_REP
Linkage SSO Monthly Event Report
ICS_LNK_SSO_MONTHLY_EVT_REP
Linkage SW Event Report
ICS_LNK_SW_EVT_REP
Linkage SW MS4 Report
ICS_LNK_SWMS4_REP
Linkage State Compliance Monitoring
ICS_LNK_ST_CMPL_MON
Linkage Federal Compliance Monitoring
ICS_LNK_FEDR_CMPL_MON
DMR Program Report Linkage
ICS_DMR_PROG_REP_LNK
Linkage Biosolids Report
ICS_LNK_BS_REP
Linkage SW Event Report
ICS_LNK_SW_EVT_REP
Permit Identifier
ICS_BASIC_PRMT_KEY_ELEM
Final Order Violation Linkage
ICS_FINAL_ORDER_VIOL_LNK
Permit Schedule Violation
ICS_PRMT_SCHD_VIOL
Discharge Monitoring Report Violation
ICS_DSCH_MON_REP_VIOL
Discharge Monitoring Report Parameter Violation
ICS_DSCH_MON_REP_PARAM_VIOL
Single Events Violation
ICS_SNGL_EVTS_VIOL
Enforcement Action Violation Linkage
ICS_ENFRC_ACTN_VIOL_LNK
Permit Schedule Violation
ICS_PRMT_SCHD_VIOL
Compliance Schedule Violation
ICS_CMPL_SCHD_VIOL
Discharge Monitoring Report Violation
ICS_DSCH_MON_REP_VIOL
Discharge Monitoring Report Parameter Violation
ICS_DSCH_MON_REP_PARAM_VIOL
Single Events Violation
ICS_SNGL_EVTS_VIOL

Violations



Program Reports



Response Files

Appendix B – Flow Startup Considerations

This section to be completed.