Photos courtesy of USDA NRCS.

NetDMR Deployment and Maintenance Guide

Version 1

October 17, 2008

CONTENTS

Page

1.0 Introduction

The Environmental Council of States, the Texas Commission on Environmental Quality, 12 states, EPA's Office of Environmental Information, and EPA's Office of Enforcement and Compliance Assurance partnered under an EPA Challenge Grant to design, develop, and demonstrate NetDMR. NetDMR is a web-based application that can be used to allow facilities that hold permits for discharges into the waterways of the United States to meet monitoring reporting requirements. The permits these facilities hold are referred to as National Pollutant Discharge Elimination System (NPDES) permits, and typically require submission of discharge monitoring reports (DMRs) to a regulating agency. NetDMR provides for submission of electronic DMRs (eDMRs) via an Exchange Network 1.1 compliant node (see www.exchangenetwork.net) to a local database or to EPA's data system for discharge information, the Integrated Compliance Information System (ICIS)-NPDES database NPDES permits are issued under the authority of the Clean Water Act.

1.1 About This Guide

This Guide is written for Administrators experienced with the configuration, maintenance, and operation of web, database, and application servers consistent with the architecture described in Section 2. Detailed instruction on the set up and implementation of each server is not provided. This Guide requires that your organization has:

(1) A fully functioning Node 1.1 compliant node.
(2) A thorough understanding of the Exchange Network (see www.exchangenetwork.net) and use of the Central Data Exchange to transfer information to the U.S. Environmental Protection Agency.
(3) A valid Network Authentication and Authorization Services (NAAS) user account.
(4) Approval or plans to request approval for electronic reporting consistent with CROMERR and any other applicable Federal, State or local regulations.

It is recommended that you coordinate with appropriate staff in your permitting, enforcement, compliance, information technology, and legal departments to determine if you meet these requirements.

1.2 Copyright/License

The NetDMR license is provided below; please review carefully. By installing NetDMR software, you are agreeing to all statements, terms, conditions, and disclaimers of the license.

1.3 Version

This is the version 1.0 of the NetDMR Deployment Guide, and corresponds to the NetDMR source code version 1.0.

1.4 Additional Resources

Additional information about NetDMR can be found at the following:

- http://www.exchangenetwork.net/exchanges/water/netdmr.htm

1.5 Acronyms and Definitions

Acronym	Description and Notes
CDX	Central Data Exchange - http://www.epa.gov/cdx/
COR	Copy of Record, legally enforceable copy of DMR submission
CROMERR	Cross-Media Electronic Reporting and Recordkeeping Rule
DMR	• Discharge Monitoring Report • Required under the Clean Water Act, repots on pollutants or other properties for water discharged into rivers, lakes, streams, etc. (other water bodies)
ECOS	Environmental Council of States
eDMR	Electronic DMR system
ICIS	Integrated Compliance Information System http://www.epa.gov/compliance/data/systems/modernization/index.html ICIS, a Web-based system, enables individuals from states and EPA to access integrated enforcement and compliance and NPDES data from any desktop connected to the Internet. EPA's ability to target the most critical environmental problems will improve as the system integrates data from all media. The public can access some ICIS data through ECHO.
ICIS-NPDES	Integrated Compliance Information System - National Pollutant Discharge Elimination System http://www.epa.gov/compliance/data/systems/index.html
IIS	Internet Information Server
Instance	A customized version of NetDMR, specific to a Regulatory Authority, such as a state or Region. A NetDMR Instance is contained within a NetDMR installation.
Installation	The NetDMR application as deployed in a hosting environment. A NetDMR Installation may be comprised of multiple instances.
IPT	Integrated Project Team
J2EE	Java 2 Platform, Enterprise Edition

Acronym	Description and Notes
NAAS	Network Authentication and Authorization Services
NEIEN	National Environmental Information Exchange Network http://www.epa.gov/exchangenetwork/index.html http://exchangenetwork.net/
NPDES	National Pollutant Discharge Elimination System
OECA	EPA Office of Enforcement and Compliance Assurance http://www.epa.gov/compliance/
OEI	EPA Office of Environmental Information http://www.epa.gov/oei/
SAK	Secure Authentication Key
SSL	Secure Socket Layer
UML	Unified Modeling Language
URL	Uniform Resource Locator
XML	eXtensible Markup Language
XSLT	XML style sheet

2.0 Architecture

NetDMR is a Java web application comprised of web, application, and database servers. The system architecture is shown in Figure 1. Users (e.g., permittees) access NetDMR by navigating to a specific URL through a web browser on the user's computer. The URL is processed by the NetDMR Web Server, which forwards the request to the NetDMR Application Server. The Application Server processes the request appropriately and accesses the NetDMR Database Server as needed.

The web server handles the HTTP requests and forwards those requests to the Application Server. The application server requires JBoss to host the NetDMR application and forwards requests to the Oracle or PostgreSQL database server via the Hibernate framework. All communication between the Local User and the Web Server, and between the Application Server and the Exchange Network Node will occur over the Secure Sockets Layer (SSL).

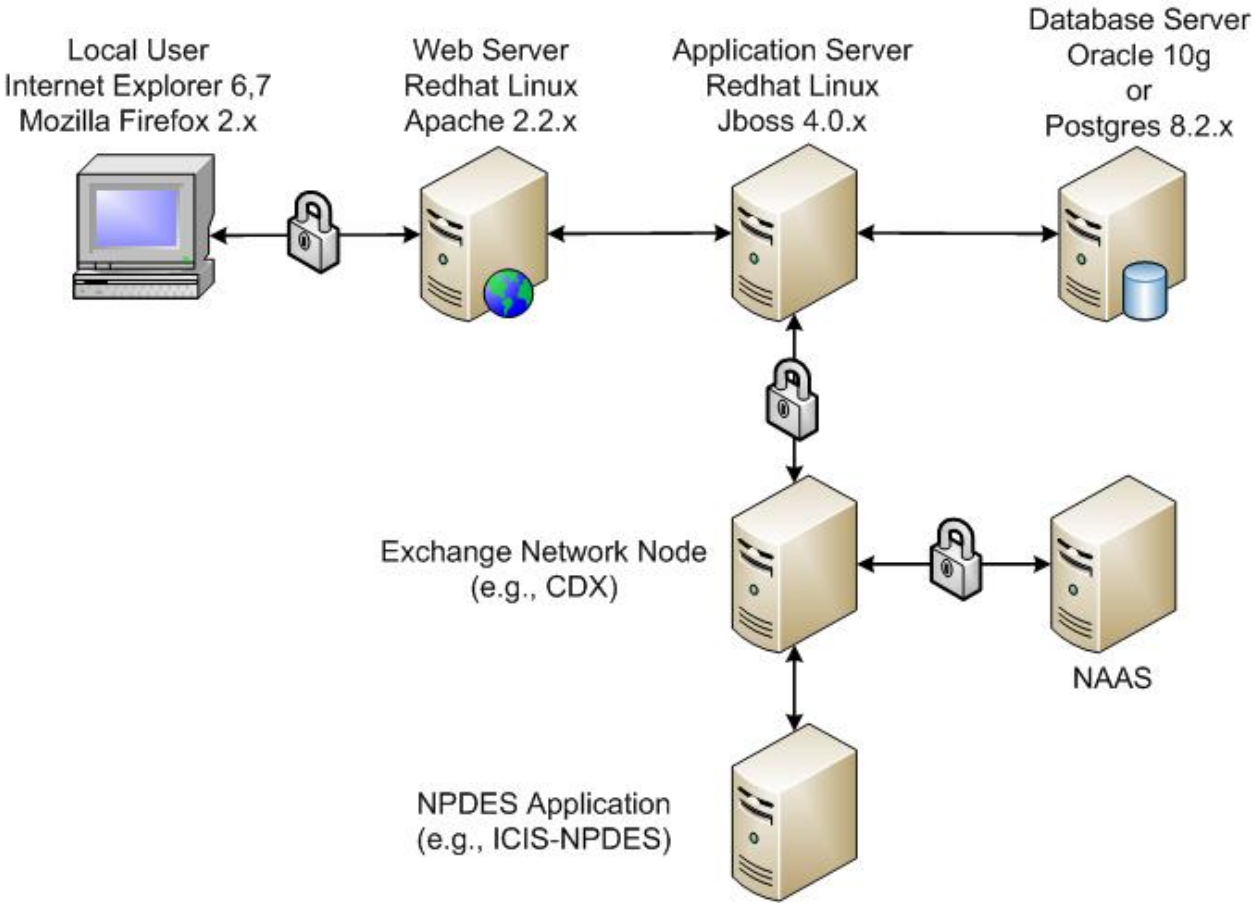

Figure 1. NetDMR Architecture

NetDMR also communicates with an Exchange Network 1.1 compliant Node (e.g., the Central Data exchange (CDX)) as shown in Figure 2. The Node provides various web services that NetDMR consumes to retrieve information from (e.g., permit information) and send information (e.g., submitted DMRs) to a NPDES application such as ICIS-NPDES. To use the services

provided by the Node, NetDMR authenticates using a valid Network Authentication and Authorization Services (NAAS) user account. For more information on the Exchange Network see http://www.exchangenetwork.net.

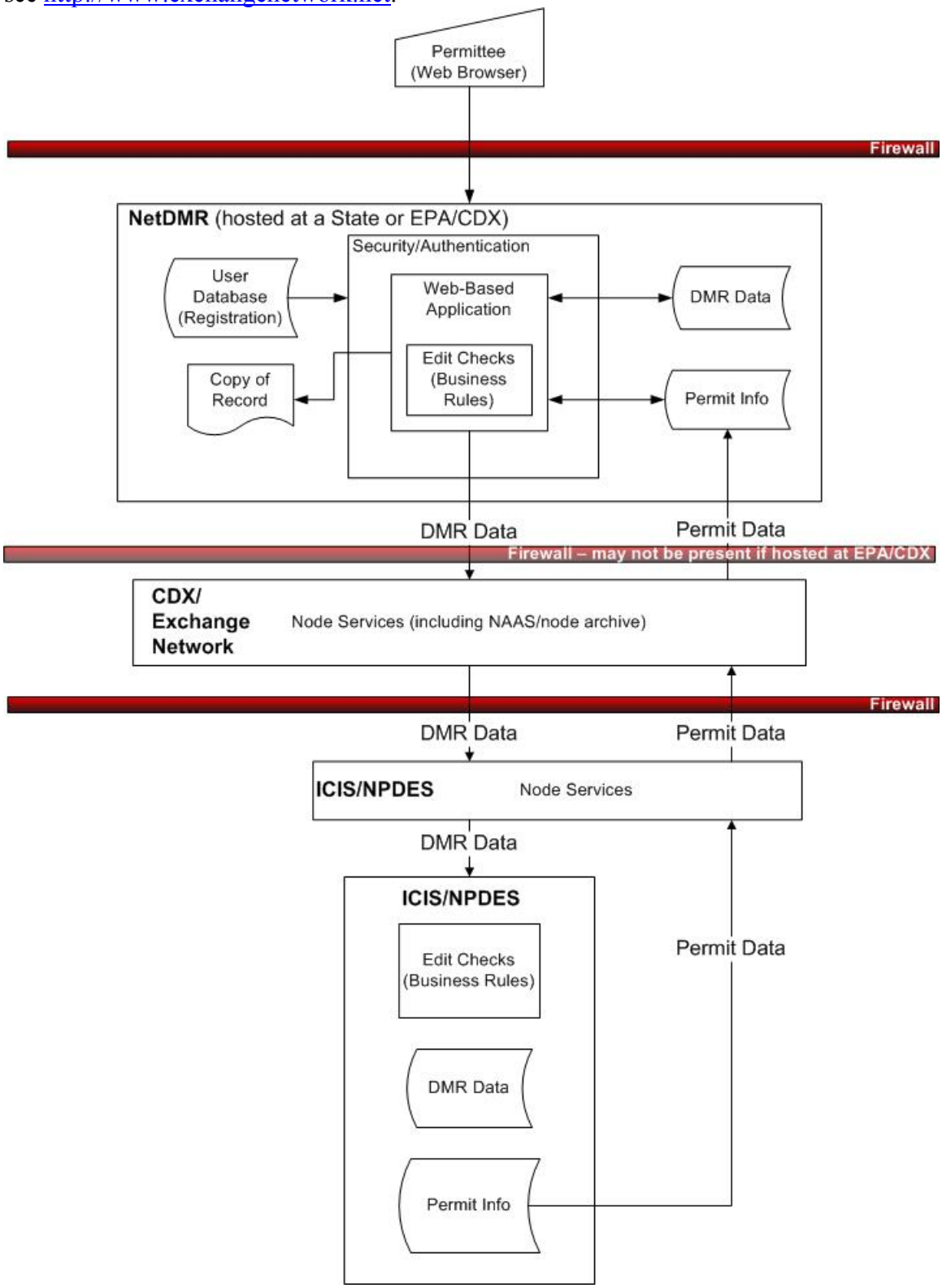

Figure 2. NetDMR Data Flows

2.1 NetDMR Installation and Instances

After installing in a properly configured hosting environment, the NetDMR application is considered a NetDMR installation. In order to make the DMR reporting and associated administrator functionality available, you must create at least one NetDMR instance. An instance is a customized implementation of NetDMR functionality for a specific regulating agency, as defined by selecting a set of geographic areas and permit types. Additional detail on creating an instance is provided in Section 7.

2.2 Exchange Network Interface

NetDMR interacts with an Exchange Network 1.1 compliant Node (e.g., CDX) to retrieve permit, empty slot, and error message information from a NPDES system (e.g., ICIS-NPDES or a state NPDES system). NetDMR also send DMRs that have been submitted using NetDMR to the NPDES system via a Node. Several Exchange Network Integrated Project (IPT) teams have been formed to specify the interaction between a Service User (e.g., NetDMR) and a Service Provider (e.g., CDX). NetDMR uses these specifications to consume the services.

2.2.1 Network Authentication and Authorization Services (NAAS)

NetDMR must have a valid NAAS account to use the services provided by an Exchange Network Node (e.g., CDX).

2.2.2 Basic Permit Data Flow

NetDMR uses the Basic Permit Data Flow (BPDF) to retrieve the list of permits that are applicable for each NetDMR instance. The list of permits is used to allow NetDMR users to request access to perform various functions for the permits (e.g., view CORs, edit DMRs). The four services that are available to NetDMR include:

- Authenticate;
- NetDMR.GetBasicPermitInfo_v1.0 (Solicit);
- GetStatus; and
- Download.

See the IPT documentation at http://www.exchangenetwork.net/exchanges/water/netdmr.htm for more information on the individual services such as the required parameters, result format, and choreography of these services.

2.2.3 Empty Slot Data Flow

NetDMR uses the Empty Slot Data Flow (ESDF) to retrieve empty slots (i.e., blank DMRs) that are applicable to each NetDMR instance. Each request is specific to an instance. NetDMR users access the on-line application interface to complete, sign, and submit the DMR. The specifications outline five services that are available to NetDMR.

1. Authenticate
2. NetDMR.GetScheduledDMRsByDate_v1.0 (Solicit)
3. NetDMR.GetScheduledDMRsByDMR_v1.0 (Solicit)
4. GetStatus
5. Download

See the IPT documentation at http://www.exchangenetwork.net/exchanges/water/netdmr.htm for more information on the individual services such as the required parameters, result format, and choreography of these services.

2.2.4 ICIS-NPDES Batch Flow

NetDMR submits Discharge Monitoring Reports (DMRs) that have been signed using NetDMR to a node 1.1 compliant endpoint, such as CDX. CDX then forwards the DMRs to ICIS-NPDES. NetDMR completes these submissions using the ICIS-NPDES Batch flow, a collection of web services and supporting processes running in ICIS-NPDES and EPA's Central Data Exchange (CDX).

See the IPT documentation at http://www.exchangenetwork.net/exchanges/water/netdmr.htm for more information on the individual services such as the required parameters, result format, and choreography of these services.

2.2.5 Error Message Data Flow

The Error Message Data Flow (EMDF) defines a specific result document to be returned to NetDMR from the ICIS-NPDES Batch Flow. The result document includes any errors that were encountered while processing the submission.

See the IPT documentation at http://www.exchangenetwork.net/exchanges/water/netdmr.htm for more information on the individual services such as the required parameters, result format, and choreography of these services.

2.3 User Environment Interface

NetDMR requires that users access the web site using one of the following supported internet browsers.

- Internet Explorer 6.x;
- Internet Explorer 7.x; or
- Mozilla Firefox 2.x.

NetDMR functionality and performance was neither tested nor validated using other browsers. Users must also have JavaScript enabled in their browser.

2.4 UML Diagrams

The following UML diagrams provide additional detail on NetDMR artifacts.

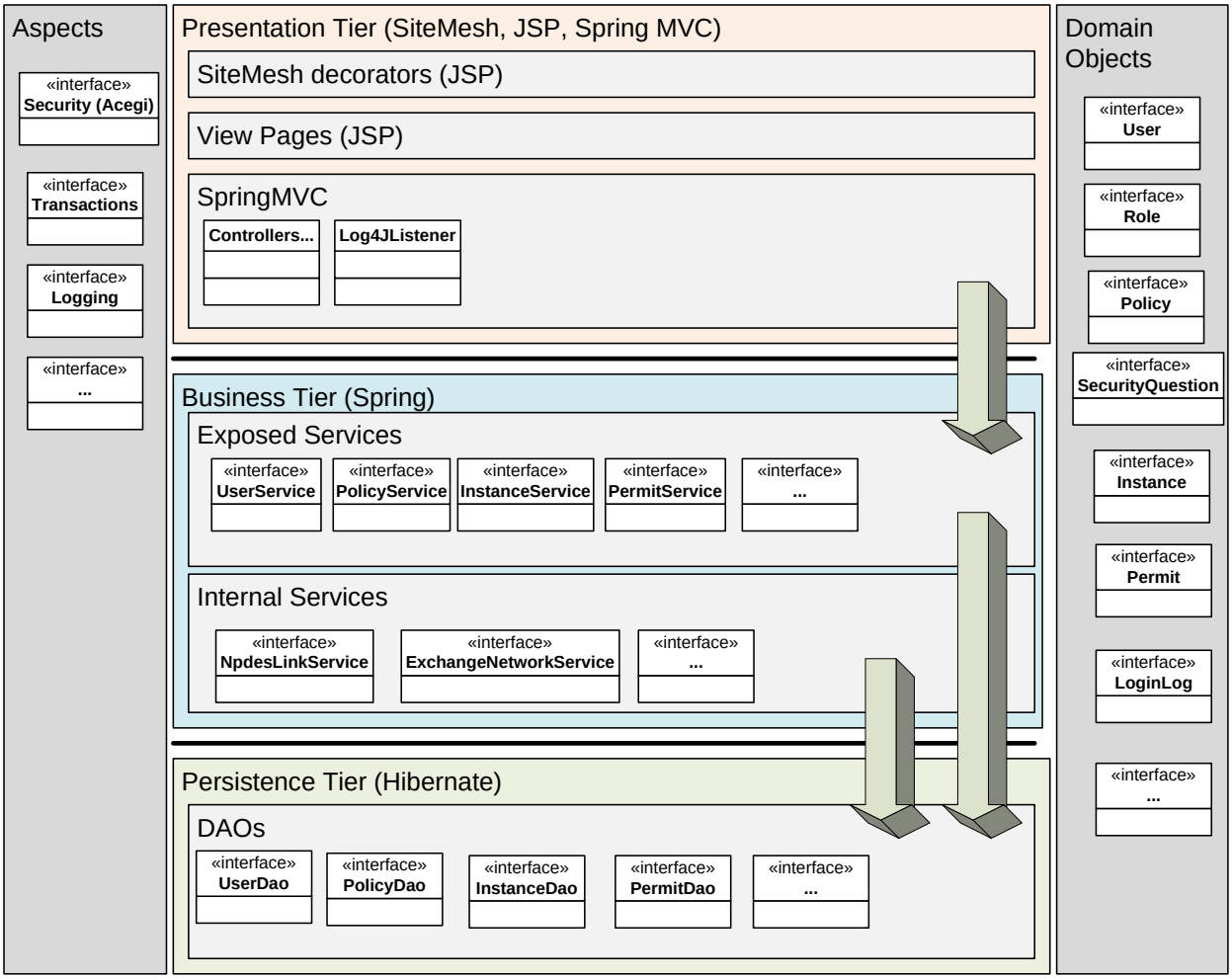

Figure 3. Overall Architecture with Example Objects

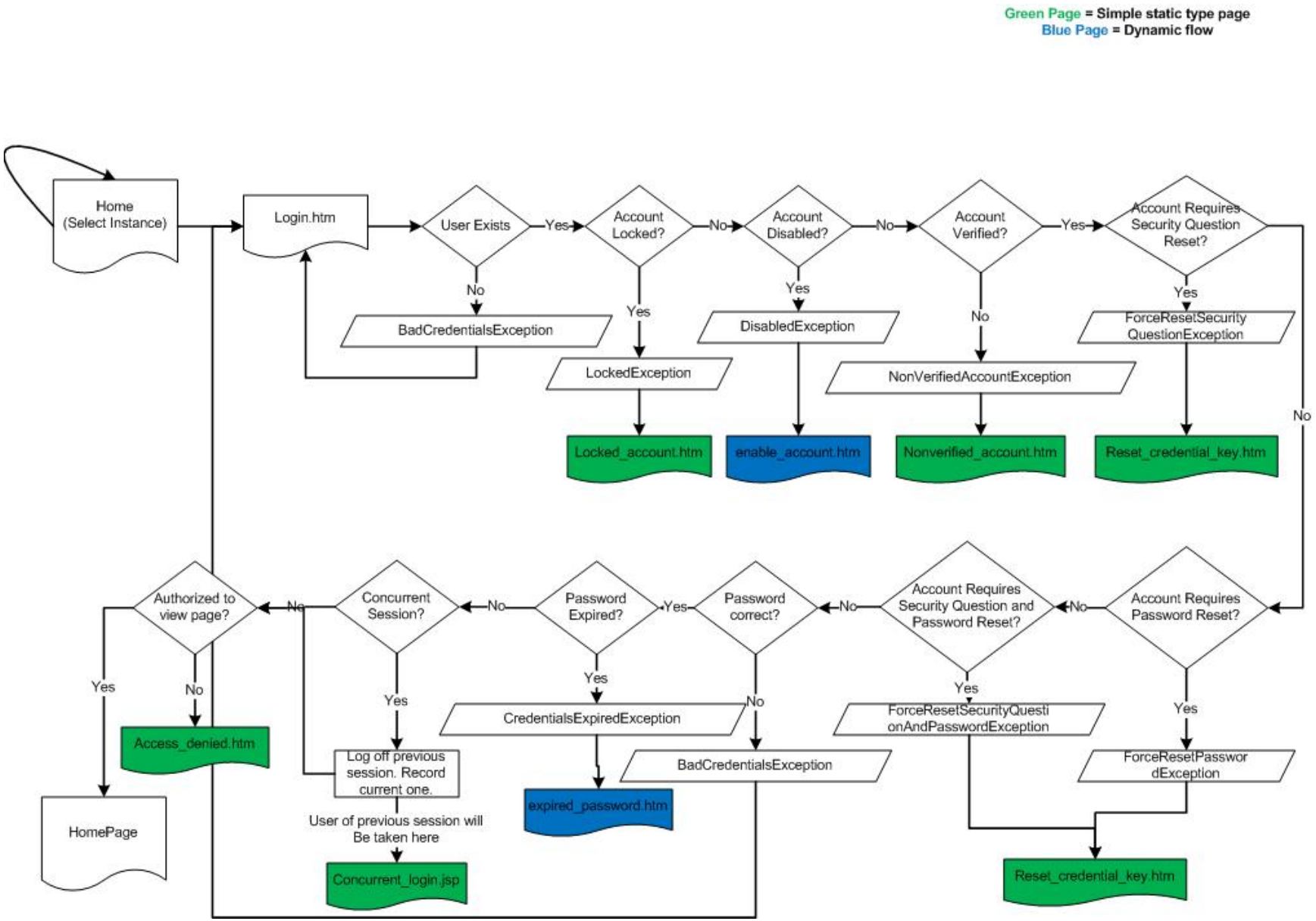

Figure 4. Login Flow

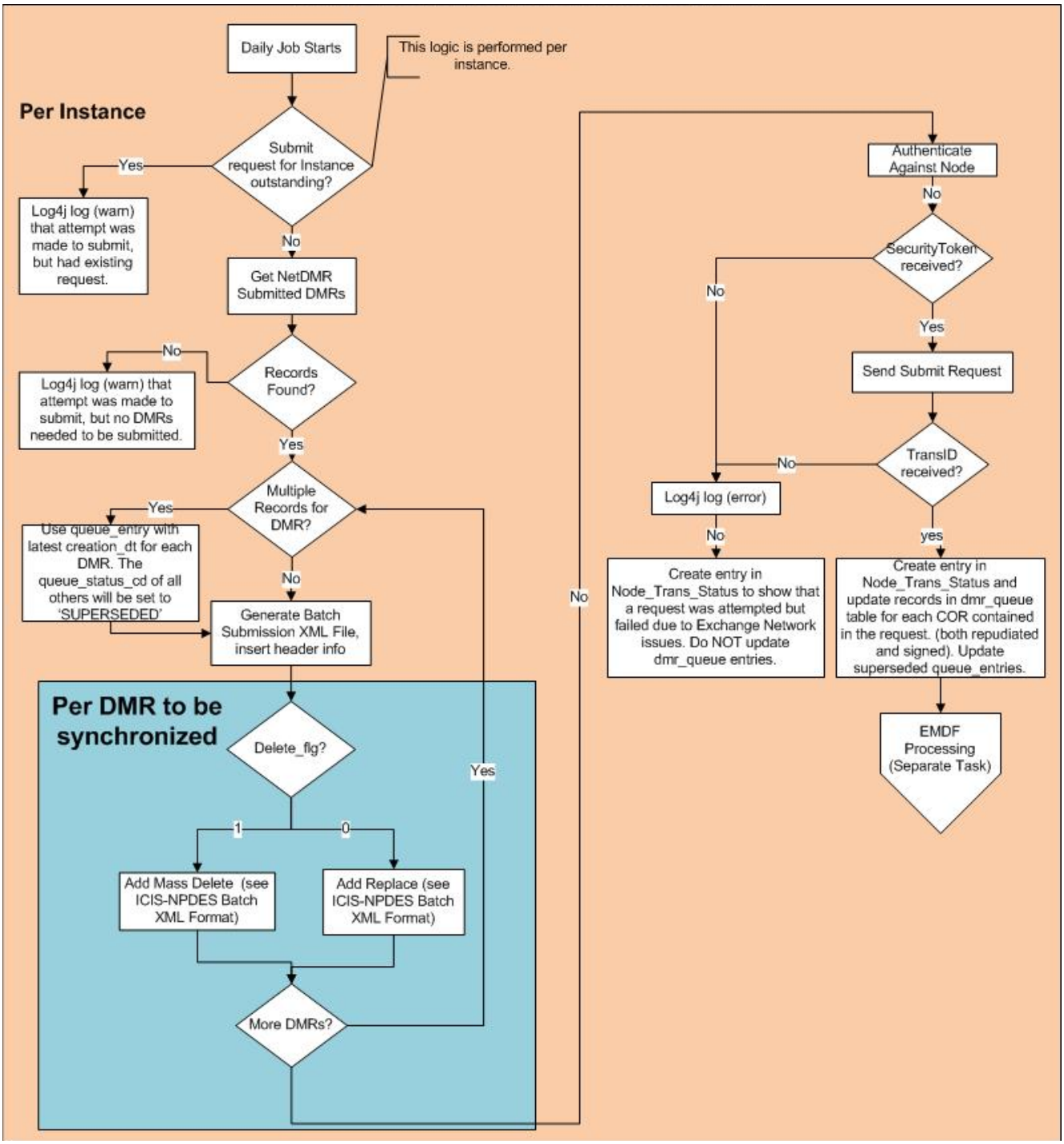

Figure 5. DMR Submittal Process

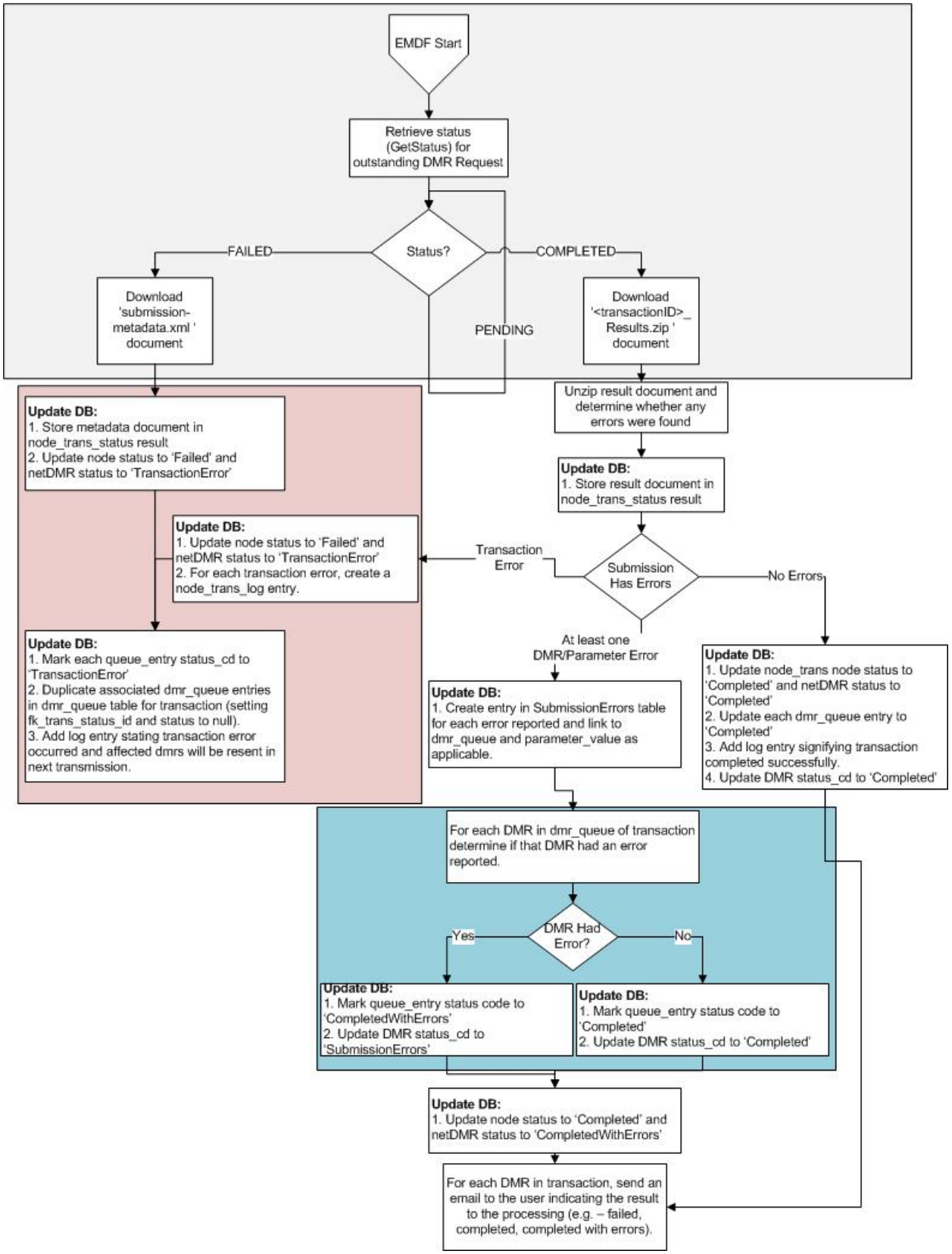

Figure 6. EMDF Process

3.0 Security

This section outlines the security standards, specifications, and algorithms that are used in NetDMR.

Type	Standard	Description
Secure Sockets Layer (SSL) specification	SSL v3 or TLS v1 [REQ 125]	Encrypts the communication between the client browser and the NetDMR web server.
Digital Certificate	x.509 Certificate with RSA 1024 bit keys [REQ 122]	The private key accompanying this certificate will be used for signing NetDMR CORs. The certificate can be used to verify the signature.
Hash Algorithm	SHA-256 [REQ 121]	An algorithm that turns a variable-sized amount of text into a fixed-sized output (hash value). NetDMR uses hashing to protect original text from discovery (e.g., passwords, secret answers) and to generate a reproducible "fingerprint" to verify information has not been changed (e.g., within Copy of Record).
Signature Algorithm	SHA256withRSA	SHA256withRSA specifies that the plaintext should be hashed using the SHA256 algorithm, and then sign the hash with the RSA algorithm.
Random Number Algorithm	SHA1PRNG	The SHA1PRNG is an algorithm for generating pseudo random numbers. This will be used by NetDMR when a random number is required (e.g., verification keys, password salts).

3.1 Application Security

All NetDMR web pages are classified as protected or publicly available. Protected pages are accessible only over Secure Sockets Layer (SSL) protocol v3 or Transport Layer Security v1.0 and require that the user provide the user name and password associated with their account for verification by NetDMR prior to allowing accessing. Protected pages are further classified to limit which authenticated users can access the page.

3.2 Certificates

The appropriate SSL configuration depends on the environment in which NetDMR is deployed (i.e., web server). SSL certificates can be purchased from vendors such as Verisign (http://www.verisign.com/) and Digicert (http://www.digicert.com/).

Similar to SSL certificates, digital certificates used for signing documents can be provided by numerous vendors.

NetDMR uses an RSA 1024 bit asymmetric key pair to sign CORs. An asymmetric key pair uses a pair of cryptographic keys - a public key and a private key. The private key is kept secret, while the public key may be widely distributed. The keys are related mathematically, but one key can not be practically derived from the other key. A message signed with NetDMR's private key can be verified by anyone who has access NetDMR's public key, thereby proving that NetDMR signed it and that the message has not been tampered with. This is used to ensure the authenticity of the signature (http://grouper.ieee.org/groups/1363/).

3.3 Hash, Signature, and Random Number Algorithms

Implementations of the specified hash, signature, and random number algorithms are available from vendors such as Sun, IBM, and The Legion of the Bouncy Castle. NetDMR can be configured to use any of these implementations, provided they implement the specified algorithm, without changing any NetDMR code. This is accomplished by registering the implementation, referred to as a Provider, through the use of Java Cryptography Extensions (JCE). More information about this process can be found at: http://java.sun.com/j2se/1.5.0/docs/guide/security/jce/JCERefGuide.html.

The Sun Provider is registered by default in the Sun JVM, and is the Provider used by NetDMR.

3.4 NetDMR and the Exchange Network

NetDMR must have a Network Authentication and Authorization Services (NAAS) account to send and retrieve information over the Exchange Network. The account must have the appropriate permissions on the application Exchange Network Node to call the required services. The list of services are defined through the Permit, ICIS-NPDES Batch, and Error Message Integrated Project Teams. A Secure Authentication Key (SAK) must be created for this account. See http://www.exchangenetwork.net/node/dev_toolbox/sak.htm for more information on SAKs. The account information will be stored within a NetDMR configuration file.

All NetDMR communications with the Exchange Network occur over SSL.

3.5 Security References

- 830-1993: IEEE Recommended Practice for Software Requirements Specifications
- Cross-Media Electronic Reporting Rule, 40 CFR Part 3
- Federal Information Processing Standards (FIPS)-approved algorithms for generating Message Digest (http://www.csrc.nist.gov/CryptoToolkit/tkhash.html)

- FIPS-approved algorithms for generating/verifying digital signatures (http://www.csrc.nist.gov/CryptoToolkit/tkdigsigs.html)
- NIST Hash Function Policy (http://www.csrc.nist.gov/pki/HashWorkshop/NIST%20Statement/NIST_Policy_on_HashFunctions.htm)
- SecureRandom Specification (http://java.sun.com/j2se/1.4.2/docs/api/java/security/SecureRandom.html)
- Security Salts (http://msdn.microsoft.com/msdnmag/issues/03/08/SecurityBriefs/)
- Spring Security module (http://www.acegisecurity.org/)

4.0 Server Requirements Planning

In planning a NetDMR deployment, consider the following factors:

- Number of instances that the installation will host
- Number of permits that the installation will host.
- Number of DMRs expected to be submitted monthly.
- Average number of parameters per permit
- Total annual permitted-parameters for your installation
- Total DMRs expected to be submitted annually
- Expected distribution pattern for DMR reporting throughout the month (based on DMR due dates)
- Size of current DMR report repository on an annual basis.
- Number of years of DMR data/copies of record you wish to remain readily accessible.
- Internal data retention policies and requirements.
- Internal document management policies and requirements.
- Redundancy requirements.

After collecting the above information, you should plan you server configuration carefully, perform testing, and monitor performance to assure that available resources are sufficient given your internal redundancy factors.

4.1 NetDMR Testing Environment

NetDMR was tested by a significant user acceptance testing team over a several month period. A summary of key statistics in the testing environment include the following:

Factor	Value
Number of Instances	15
Total Permits Retrieved	39,358
Total DMRs Retrieved	25,128
Maximum DMR/Empty Slot Retrievals in a One Day Period	1,138
Maximum Basic Permit Retrievals in One Day	101
Maximum DMR Batches Submitted in One Day Period	16
Total Parameters across all DMRs	269,924
Total DMRs Submitted	743
Unique User Accounts	356
Total Successful Logins During the Entire Testing Period	6,197
Maximum Number of Logins in a One Hour Period	191
Maximum Number of DMRs Signed in a One Hour Period	332
Maximum Number of DMRs Submitted in a Single Submission (submissions are per instance)	83

The web server environment that supported the above activity is as follows:

Environment Specification	Value
Type of Server	Dell Servers Shared environment 8 - 2.8 GHz Processors 4 - 8 GB RAM
Number of Servers	2
Server OS	Windows 2003 Standard and Enterprise Server, Service Pack 2
Memory Allocation to NetDMR	1.0 GB
Maximum memory utilization during stable testing periods	748 MB
Clustered environment?	Yes, sticky sessions enabled
Environment shared with other applications?	Yes

The database server environment that supported the above activity is as follows:

Environment Specification	Value
Type of Server	Shared Environment Clustered 2 nodes, 4 dual core CPUs per node
Number of Servers	2
Server OS	Linux rhel4
Database	Oracle 10g (10.2.0.4)
Storage space allocation	2 GB
Clustered environment?	Yes, 10g RAC
Environment shared with other applications?	Yes

5.0 Prerequisites

NetDMR is a Java web application that requires web, application, and database servers for proper operation. Figure 1 provides the System Architecture that must be in place before you can deploy NetDMR and includes:

- Web server: Apache 2.2 or IIS version 7
- Application server: JBoss 4.0 or 4.2
- Database: Oracle 10g or PostgreSQL 8.2

The web server handles the HTTP requests and forwards those requests to the application server. The application server requires JBoss to host the NetDMR application and forwards requests to the Oracle or PostgreSQL database server via the Hibernate framework. All communication between the Local User and the Web Server, and between the Application Server and the Exchange Network Node will occur over the Secure Sockets Layer (SSL).

5.1 Resource Prerequisites

The following resources are required for NetDMR installation:

- Web server: Apache 2.0 or IIS version 7, or Tomcat/JBoss capable of forwarding *.htm requests for the application to the application server
- Application server: JBoss 4.0 or 4.2
- Java EE 1.4
- Database: Oracle 10g or PostgreSQL 8.2
- Secure Socket Layer (SSL) capability
- Valid signing certificate, including an RSA 1024 bit asymmetric key pair in the PKCS12 format (see http://www.rsa.com/rsalabs/node.asp?id=2138) stored in an keystore.
- A database schema to store NetDMR application tables
- A user account with read/write access to the database
- A JBoss datasource
- Standard zip software
- An externally accessible web address for NetDMR access
- An email server through which NetDMR can send email
- A node endpoint for a 1.1 compliant Node that implements the Exchange Network services (see http://www.exchangenetwork.net/node/node1.1.htm)
- A valid NAAS account for NetDMR authentication and communications with the Exchange Network. The Exchange Network Help Desk can provide more information on requesting and obtaining a NAAS account (see http://www.exchangenetwork.net/contacts/helpdesk.htm).
- Appropriate name-value pair for use in the DMR data flow (requires coordination with EPA/OECA staff, see the EPA contact listed on http://www.exchangenetwork.net/exchanges/water/netdmr.htm

5.2 Installation and Instance Prerequisites

After NetDMR installation is completed, you will customize the application for your Agency. Prior to beginning installation, you should identify the lead program area or business area representative managing NetDMR. You should work with this representative to define the instances required and properties for each instance. You will need this information to verify proper installation of NetDMR.

5.3 Other Prerequisites

The Cross-Media Electronic Reporting Rule (CROMERR, 40 CFR Part 3) provides the legal framework for electronic reporting under all of EPA's environmental regulations. Additional information on CROMERR compliance and EPA requirements can be found at http://www.epa.gov/exchangenetwork/cromerr/.

NetDMR is designed to be compliant with the Cross-Media Electronic Reporting Rule. A CROMERR checklist has been prepared and submitted for the NetDMR instance that will be hosted by EPA headquarters. The checklist can be downloaded from http://www.exchangenetwork.net/exchanges/water/netdmr.htm.

Note that their may be additional legal requirements for your State, Region, or Territory that must be met prior to submitting a CROMERR application to EPA. You should confer with your Agency legal counsel to verify that your laws and/or regulations provide sufficient legal authority to implement electronic reporting and enforcement of affected programs.

6.0 Installing NetDMR

NetDMR was developed in a RedHat Linux environment and tested in both RedHat Linux and Windows environments. If you plan to install in a different environment, these instructions will likely need to be modified for the target environment.

A summary of the installation process is as follows:

- Step1: Complete server planning and configuration (see Section 3.0).
- Step 2: Verify that all prerequisites are completed (see Section 4.0).
- Step 3: Install the database.
- Step 4: Create the Certificate Keystore.
- Step 5: Configure and Deploy JBoss Datasource
- Step 6: Create the Initial System Administrator
- Step 7: Configure and Deploy NetDMR Application
- Step 8: Verify Proper Installation

6.1 Step 1: Complete Server Planning and Configuration

Refer to Section 3.0 for server planning and configuration considerations.

6.2 Step 2: Verify Prerequisites

Refer to Section 4.0 NetDMR Installation prerequisites.

6.3 Step 3: Install the Database

NetDMR can be installed with either an Oracle or a PostgreSQL database.
This step assumes the following:

- You have determined whether you are using Oracle or PostgreSQL.
- The required database server is properly configured.
- You and/or the database administrator completing this Step has appropriate access to the server to perform the required activities.
- You have downloaded the appropriate NetDMR database creation script file from the Exchange Network site at http://www.exchangenetwork.net/exchanges/water/NetdmrDownload.htm. The script files are named as follows:
  NetDMR_OracleSQL.v1.0.sql or
  NetDMR_PostgreSQL.v1.0.sql

6.3.1 Installing the NetDMR Database in an Oracle 10g Environment

This step assumes the following:

- You plan to use an Oracle 10g database for NetDMR.

- You have downloaded the NetDMR Oracle database creation script file (NetDMR_OracleSQL.v1.0.sql) from the Exchange Network site at http://www.exchangenetwork.net/exchanges/water/NetdmrDownload.htm.

If the preceding assumptions are accurate, follow these steps to install the NetDMR Oracle 10g database:

1. Create a schema to hold the application tables and an associated database user account (e.g. netdmruser).
2. Generate the desired table space or spaces in which you would like to store the tables and indices.
3. Open the database creation script file NetDMR_Oracle.1RC1.sql
4. Modify all the occurrences of 'netdmruser' with the name of the schema created in Step 2.
5. Assign the schema a default table space or modify the creation script to assign each table and index to the desired table space.
6. Log in as a user account created to hold the tables.
7. Run the script; the script creates the required tables, indexes, constraints, and inserts sample data.

6.3.2 Installing the NetDMR Database in a PostgreSQL 8.2 Environment

This step assumes the following:

- You plan to use a PostgreSQL 8.2 database for NetDMR.
- You have downloaded the NetDMR PostgreSQL database creation script file (NetDMR_PostgreSQL.v1.0.sql) from the Exchange Network site at http://www.exchangenetwork.net/exchanges/water/NetdmrDownload.htm.

If the preceding assumptions are accurate, follow these steps to install the NetDMR PostgreSQL 8.2 database:

1. Either determine which existing table space you desire to use for NetDMR or create a new table space.
2. Either determine which existing "database" on the desired server you wish to use for NetDMR or create a new one with the desired table space.
3. Create a schema to hold the application tables and an associated database user account (e.g. netdmruser).
4. Open the database creation script file NetDMR_PostgreSQL.1RC1.sql
5. Modify all the occurrences of 'netdmruser' with the name of the schema created in Step 1.
6. Log in as the created user from Step 3.
7. Run the script; the script creates the required tables, indexes, constraints, and inserts sample data.

6.4 Step 4: Configure and Deploy the JBoss Datasource

In order for NetDMR to connect to the schema you set up in Step 3, you will need to create a JBoss datasource as follows:

1. Open the provided netdmr-oracle-ds.xml file or the netdmr-postgresql-ds.xml file and modify the following to reflect the your environment:
   - jndi-name: You can choose any name for this, or leave the default. If you change the default value, you will need to make a corresponding change to the db.jndiURL property in the netdmr.properties file described in the next section.
   - connection-url: The connection string for the Oracle or PostgreSQL database. The '1.1.1.1' and 'sid' must be replaced with the correct server name or IP and sid value for the your database.
   - user-name: The username for the schema created in Step 3.
   - password: The password for the username created in Step 3.

2. Copy the modified netdmr-oracle-ds.xml or the netdmr-postgresql-ds.xml to the deploy directory of the appropriate JBoss server.

6.5 Step 5: Configure and Deploy the NetDMR Application

The next step is to update the configuration file in the NetDMR WAR file to reflect your environment. Download the WAR from the Exchange Network website at:

http://www.exchangenetwork.net/exchanges/water/NetdmrDownload.htm

1. Download and unzip the war file. WAR files are simply zip files renamed to 'war' and can be unzipped with any standard zip program (e.g., Winzip). If the program does not recognize the file as a valid zip file, rename the file to netdmr-web.zip.

2. Open the file netdmr-web/WEB-INF/classes/netdmr.properties. The properties listed in Table 6-1 should be updated or verified as appropriate.

3. After updating the properties, rezip the contents of the WAR file and name the file netdmr-web.war.

4. Verify that the new WAR file directory structure is exactly the same as the original.

5. Copy the newly created WAR file to the deploy directory of your JBoss server.

6. Start the JBoss server, if it is not already started.

Table 6-1. NetDMR Configuration Properties			
Group	**Name**	**Description**	**Default**
Access URL	netdmr.base.url	The https URL at which the application will be deployed (e.g., https://www3.tceq.state.tx.us/netdmr-web). The netdmr-web is the same as the WAR file name. If an SSL enabled environment is not available, NetDMR will work correctly over simple http.	netdmr-web
Messaging Properties	email.server	The SMTP server through which email will be sent.	Default not available
	email.port	The port over which emails are sent.	25
	email.server.username	If required, the user to authenticate to the email server	Default not available
	email.server.password	If required, the password to authenticate to the email server	Default not available
	email.from	The information that will be listed in the 'from' line of emails generated by NetDMR	NetDMR-XX
Hibernate Dialect	hibernate.dialect	Setting which tells Hibernate which type of database the application is accessing.	For PostgreSQL: org.hibernate.dialect.PostgreSQLDialect
			For Oracle: org.hibernate.dialect.Oracle10gDialect
JBoss Datasource Properties	db.schema	The database schema that holds the NetDMR tables	
	db.jndiURL	The JNDI url to the database. This reflects the name you chose for your JBoss datasource.	
Exchange Network Authentication Credentials	authenticate.username	The username for Exchange Network Authenticate requests	Default not available
	authenticate.credential	The credential for Exchange Network Authenticate requests	Default not available
	authenticate.method	The authentication method for Exchange Network Authenticate requests	password
	exchange.network.xsd	URL for the schema of the node.	http://www.w3.org/2001/XMLSchema
	exchange.network.node	URL for the node connection.	http://www.ExchangeNetwork.net/schema/v1.0/node.xsd
Scheduled	status.cron.expression	The frequency that the system should request from	0 30 * * * ? *

Table 6-1. NetDMR Configuration Properties			
Group	**Name**	**Description**	**Default**
NODE Interactions		the NODE the status of outstanding requests.	
	solicit.cron.expression	The frequency that the system will scan the queue and send pending requests to the NODE.	0 15 * * * ? *
Basic Permit Data Flow (BPDF) Configuration Properties	bpdf.node.endpoint	The node endpoint to use for BPDF Exchange Network requests	
	netdmr.solicit.bpdf.request	The name of the BPDF request service	NetDMR.GetBasicPermitInfo_v1.0
	prepare.bpdf.solicit.cron.expression	A CRON expression that specifies when a BPDF refresh should be made of all the instances.	0 0 4 ? * * (i.e., trigger at 4 am every day)
Empty Slot Data Flow (ESDF) Configuration Properties	esdf.node.endpoint	The node endpoint to use for ESDF Exchange Network requests	
	prepare.esdf.solicit.cron.expression	A CRON expression that defines how often the permits are checked to determine if an ESDF solicit request should be made for the permit.	0 5/15 * * * ? (i.e., trigger every 15 minutes starting at 5 past the hour)
	esdf.mpsd.start.date	One of the properties that determines the set of DMRs that will be retrieved for a permit. How many months in the past (from the current date) the monitoring period start date empty slots should be retrieved on the initial retrieval for a permit.	-12
	esdf.mpsd.end.date	One of the properties that determines the set of DMRs that will be retrieved for a permit. Initial Retrieval: How many months in the future (from the current date) the monitoring period start date empty slots should be retrieved Subsequent retrievals: The buffer of DMRs whose	1

Table 6-1. NetDMR Configuration Properties			
Group	**Name**	**Description**	**Default**
		monitoring period has not yet started, but will within the specified number of months specified in this property..	
	esdf.mped.end.date	How many months in the future (from the current date) the monitoring period end date empty slots should be retrieved on the first retrieval The number of months from the current date that the initial pull of DMRs should go into the future for the MPED date range.	2
	esdf.mped.refresh.epoch	The esdf.mpsd.end.date property defines a buffer in which DMRs that have not yet started are downloaded. This property defines the number of days before the buffer 'runs out' that a request for the next set of DMRs of a permit will be made. For example, a value of 7 would mean that the next set of DMRs should be pulled 7 days before the end of the buffer.	
	netdmr.solicit.esdf.request.dmrByDate	The name of the ESDF request DMRs by date service.	NetDMR.GetScheduledDMRsByDate_v1.0
	netdmr.solicit.esdf.request.dmrByDmr	The name of the ESDF request DMRs by DMR service.	NetDMR.GetScheduledDMRsByDMR_v1.0
DMR Submission Configuration Properties	emdf.node.endpoint		http://testngn.epacdxnode.net/cdx-enws10/services/NetworkNodePortType_V10
	dmr.node.endpoint	The Web Service endpoint to which DMRs should be submitted and EMDF results retrieved.	http://testngn.epacdxnode.net/cdx-enws10/services/NetworkNodePortType_V10
	submit.dmr.cron.expression	CRON expression for how often the DMR submission process will run. Only a single submission can be outstanding (being processed by ICIS) per instance.	0 30 * * * ? (i.e., trigger every hour at 30 minutes after the hour)

Table 6-1. NetDMR Configuration Properties			
Group	**Name**	**Description**	**Default**
	dmr.submit.icis.user.id	The ICIS User ID that will be included in the header of the DMR submissions	NetDMR
	dmr.dataflow	The dataflow parameter for the submit web service used to submit DMRs.	ICIS-NPDES2
	dmr_XmlSourceName	Name portion of name/value pair used to designate DMR source.	Source *Note: must be valid source name in ICIS XML/DTD
	dmr_XmlSourceValue	Value portion of name/value pair used to designate DMR source.	NetDMR *Note: must be valid source value in ICIS XML/DTD
KeyStore information	keystore.location	Location of certificate pkcs12 keystore that holds private keys and certificates for signing DMRs that have been submitted	
	keystore.password	The password for accessing the keystore. The same password must be used for both the keystore itself and any certificates stored within it.	
Suspect log analysis	suspectLogAnalysis.cron.expression	CRON expression detailing when checks should be made to determine if the suspect logs need to be run.	0 0 0 * * (i.e., trigger at midnight each day)
Import DMR process	importDmr.cron.expression	CRON expression for how often the import process will run.	0 3/15 * * * ? (i.e., trigger every 15 minutes starting at 3 past the hour)
Nightly Clean Up process	nightlyCleanUp.cron.expression	CRON expression to purge partial records from COR table. When a user starts the sign and submit process, NetDMR generates a Data XML Blob and stores it as a new record in the COR table. If the user signs and submits, NetDMR updates this record with the full COR. If the user aborts the sign and submit process, the full COR is never generated and the partial COR is unnecessary. The nightly clean up process deletes these records every evening.	0 0 3 * * ? (i.e., trigger at 3 am every day)

6.6 Step 6: Create an Initial System Administrator

A System Administrator user is required to login and complete configuration of the NetDMR installation, create NetDMR instances, and perform some customization of the instances. To create the initial System Administrator, follow the steps below. Note that you will need to use an account that has read and write access to the schema created in Step 2.

1. Access the schema created in Step 2.
2. Access the Installation table
3. Change the allowsysadmincreate_flg integer value from the default of '0' to '1'. This will allow access to the system administrator account creation page.
4. Change the adminkeycode_txt character varying(64) to string you would like to use as the verification key on the System Administrator creation page.
5. Commit the database changes
6. Open your browser
7. Access the following: http://<NetDMR URL you specified in Step 5/public/sysadmin_create.htm
8. Enter the required information in the Create New Account as a System Administrator section.
9. Click *Submit*.
10. On the Verify System Administrator Request page, enter the verification key string.
11. Click *Submit.*

You can then log in to NetDMR as a System Administrator and continue the installation and configuration process.

6.6.1 Lock the System Administrator Creation Functionality

It is recommended that the number of users serving as System Administrators for a NetDMR installation be limited to a small number. System Administrators have significant access to NetDMR settings and instances. You can lock the availability of the System Administrator Creation page as follows:

1. Open your browser
2. Access the following: http://<NetDMR URL you specified in Step 5>/sysadmin_create.htm
3. In the Lock Installation for System Administrator Creation section, click to check the box next to 'Lock Creation'.
4. Click *Submit*.
5. On the ***Verify System Administrator Request*** page, enter the verification key string.
6. Click *Submit.*

If you would like to reactivate this page, repeat the activities listed in Step 6.

6.7 Step 7: Create the Certificate Keystore

NetDMR requires the use of digital certificates to securely sign copies of record and meet CROMERR requirements. The following must be completed before NetDMR can sign CORs:

1. Generate certificate (public/private key pair)
2. Add the certificate to the NetDMR keystore
3. Register the certificate for use by NetDMR

6.7.1 Generate Certificate

The key pair must be an RSA 1024 bit asymmetric key pair and must be provided in the PKCS12 format (http://www.rsa.com/rsalabs/node.asp?id=2138). PKCS12 is an open standard for storing the private key and associated public key certificate. Numerous Certificate Authorities (CAs) such as Thawte and Verisign can generate PKCS12 formatted key pairs. The Exchange Network CA may also be able to generate the key pair. Deployers can choose which Certificate Authority (CA) to use. It is also possible to generate the certificate internally and self-sign the certificate. While this is probably acceptable for the development and test environments, it is recommended that the certificate used in production is signed by a third party CA to add an additional level of verification to the certificate generation process.

6.7.2 Add Certificate to Keystore

A keystore is a file that stores public and private keys. NetDMR uses a PKCS12 keystore to store PKCS12 certificates. Keystores can, and should, be protected with a password. The file must be stored in the file system and must be accessible to the NetDMR application server. If NetDMR is deployed in a clustered environment all application servers in the cluster must have access to the same certificate. This can be accomplished by storing the keystore in a central location that is accessible to all application servers (e.g., in a SAN environment) or by duplicating the keystore on the local file system of each application server. Each deployment of NetDMR will need to determine the best approach given environment and other standards. The location of the keystore is provided to NetDMR within the NetDMR configuration file.

A member of your staff that has access to the file system must add the certificate to the keystore. There are various methods to add certificates to keystores. One option is to use the IBM KeyMan utility (http://www.alphaworks.ibm.com/tech/keyman) as it provides a graphical user interface to create keystores and manage the certificates within a keystore.

It is recommended that the permissions on the keystore file be limited to only those server users that must access the keystore. A server user is a staff member that has an account to access the physical server upon which NetDMR is deployed. It also includes accounts created for a specific service running on the server. For example, in many deployments an account is created for the web server to run under. Creating a specific account for the web server allows the permissions associated with the account to be the bare minimum required for the service to operate under. The management and use of server users is outside the scope of NetDMR.

6.7.3 Register Certificate

After the certificate has been added to the keystore, a NetDMR System Administrator must register the key for use by NetDMR via the NetDMR System Administrator interface. The Administrator must log in to NetDMR and select which certificate in the keystore should be used to sign CORs.

When a certificate is registered for use in NetDMR, the alias and public key of the registered certificate, as well as the date and time the certificate was registered, are stored in the NetDMR database.

6.8 Step 8: Verify Proper Installation

Complete the following three tests to verify proper installation of NetDMR:

- **Test 1**. Use an internet browser to access the URL http://<installation domain>/netdmr-web/public/home.htm where <installation domain> is replaced with the domain name or IP address of the JBoss server and the port, if necessary.

- **Test 2**. Create an Instance. Prepare the list of instance configuration options and the selections you plan to make for the new instance and account information for the first Internal Administrator for the instance. Open your browser and navigate to the NetDMR URL. On the ***Home*** page, select 'Start-up Instance' from the box next to Regulatory Authority and click the *Go* button. Login to NetDMR with the System Administrator account you created in Step 6, click **Create an Instance**, enter the information required for your instance, including the information to create the first Internal Administrator account, and save the instance settings.

- **Test 3**. Test the account preparation, basic permit data flow, and empty slot data flow.
  - Step 1: Create several test accounts. Prepare a list of the information needed to create a temporary Signatory/Permit Administrator account to user for this test. You also need to identify one permit ID that you would like to request access to for this user.
  - Step 2. Open your browser and navigate to the NetDMR URL. From On the ***Home*** page, select the new instance name from the box next to Regulatory Authority and click the *Go* button.
  - Step 3: Ask your initial Internal Administrator to login, click **Instance** under the **Manage** menu, and click *Edit Instance*. Make selections, enter subscriber agreement terms and conditions, enter a custom email note, and enter News and Notices. Save the Instance customization options.
  - Step 4: On the ***Home*** page, verify the News and Notices are displayed as expected.

- Step 5: Click the link to Register for a new account and enter the information required to create the temporary Signatory/Permit Administrator account.
- Step 6: Login to the email associated with the temporary Signatory/Permit Administrator and complete the account creation process. Verify that the email note specified by the initial Internal Administrator appears at the bottom of the account creation email.
- Step 7: Login as the Signatory/Permit Administrator, click Request Access, enter the permitID you plan to user for testing, click *Update*, select the Signatory role, and provide the information required to complete the request.
- Step 8: Print the subscriber agreement and (1) verify that the contact and mailing information for your Regulatory Authority is correct, (2) verify that the terms and conditions match the information you specified for the Instance, note the Subscriber Agreement number.
- Step 9: Login as the initial Internal Administrator, click **Process Agreements** under the **Manage** menu, enter the Subscriber Agreement Number from Step 9, and approve the signatory access request.
- Step 10. Under the **View** menu, click **Network Activity**. Check the ***Network Activity*** page over the next hour or two, and verify that the ESDF request has completed.
- Step 11. Login as the Signatory/Permit Administrator, click Search on the Search DMRs and CORs tab, and view the ***DMR/COR Search Results*** page. Verify that at least one DMR was returned for the testing permitID. If you have a hard copy of a DMR previously submitted by that permittee, select Edit DMR in the row of interest, click Go, and verify that the information shown on the ***Edit DMR*** page matches the hard copy
- Step 11. Login as the initial Internal Administrator and revoke signatory and permit administrator rights from your temporary permit administrator.

6.9 Step 9: Set Instance Status to On-line

By default, each instance you create is associated with a status of 'On-line for internal administrators only'. To make the instance available to all users, follow the steps below:

- Step 1: Open your browser and navigate to the NetDMR URL. From the ***Home*** page, click the update icon in the row for the instance you would like to make available to external users.
- Step 2. From the ***Edit Instance*** page, click to set the Instance Status to 'On-line' and click Save.
- Step 3: Click Save on the ***Confirm Edit Instance*** page to verify your changes.

6.10 Troubleshooting and Known Issues

- If you plan to deploy NetDMR in a JBoss 5 environment, you must first remove two library jar files from the deployment WAR file. These jar files (stax-api-1.0.1.jar, wstx-asl-3.2.6.jar) are already included in JBoss version 5 and will cause a conflict during deployment if they remain in the NetDMR WAR file.

- After deploying NetDMR, if an adjustment to the properties file is needed, you must restart the application for the properties to take effect.

- If the you have started NetDMR and find that you need to make an adjustment to one of the CRON job setting in the properties file, you must clear the content of all QRTZ* tables, adjust the properties files, and restart the application for the change to take effect.

7.0 Customizing NetDMR After Installation

NetDMR System Administrators configure a NetDMR installation, create and manage all NetDMR instances that are part of that installation, and manage installation settings.

A NetDMR Installation is the NetDMR application as deployed in a hosting environment. A NetDMR Instance is a customized version of NetDMR, specific to a Regulatory Authority, such as a state or Region.

A single NetDMR installation can include one or more instances. An installation is analogous to enterprise email server software, and each NetDMR instance associated with the installation is analogous to an email account. Global settings, such as data flow request frequency and timing, are specified at the installation level. Instance settings, such as Agency Maps and password change frequency, and are specified at the account level.

7.1 Customizing NetDMR Using the System Administrator Interface

System Administrators can customize installation settings and create instances.

7.2 Installation Settings

Follow the steps below to view and edit installation settings:

1. Login to NetDMR with the System Administrator Account
2. Click **Setting** under the **Manage** menu on the ***Home*** page.
3. Click *Edit Settings* on the ***View Settings*** page.
4. Click in the box next to User/COR Purge Schedule to change the setting.
5. Click in the box next to Time Zone to change the setting.
6. Click in the box next to Suspect Analysis Frequency to change the setting.
7. Click in the box next to Suspect Analysis Log Data Subset to change the setting.
8. In the NetDMR Signing Certificate table, click in the Select Certificate column to specify the active signing certificate.
9. Click *Save*.
10. Click *Save* again on the ***Confirm Edit Settings*** page to finalize your modifications.

7.3 Create Instances

An instance is a customized version of NetDMR, specific to a Regulatory Authority. Follow these directions to create a new instance in NetDMR and specify general information, create the initial internal administrator, assign Agency Maps, and update the subscriber agreement.

Create General Information

1. Log into NetDMR with the System Administrator account.
2. Click Create New Instance to access the ***Create Instance*** page.
3. On the ***Create Instance*** page:

- Enter the name of the instance.
- Select the time zone.
- Select the number of security questions users must answer when creating an account.
- Select how often users will be required to change their password.
- Select whether to allow multiple signatory requests in a subscriber agreement.
- Upload the graphic that will be shown in the top right corner of each page by clicking *Browse* and selecting it from your hard drive. Graphics must be in the *.jpg format. The recommended size specifications are:
  - Height: 60 pixels
  - Width: 20 pixels

  Graphics of a different size will likely appear distorted.
- Specify the Submittal Postal Code. The DMR data flow requires that NetDMR include in the submission file name a submittal Postal Code. NetDMR uses the code you provide to properly name all DMR submission files that are submitted to CDX and ICIS. You will need to coordinate with EPA ICIS staff on the proper postal code for each instance you create.
- Specify a Contact Name. This name will be displayed on the NetDMR Contact Information page.
- Specify a Contact Email. This email will be displayed on the NetDMR Contact Information page and the Home page.
- Specify a Contact Phone. This phone number will be displayed on the NetDMR Contact Information page.

Specify Agency Maps

4. On the ***Create Instance*** page, specify the Agency Maps by selecting the states or areas appropriate for this instance. You should consult with the program area or business lead for the Instance to determine appropriate Agency Maps.
   - Select the permit prefix.
   - Select the agency type code. This is the code used in ICIS-NPDES to categorize the agency that administers the permit. Options, as of August 2008, are as follows:

Agency Type Code Listing		
CNT : County	IN4 : Interstate	SC1 : State Contractor
CT1 : County	IN6 : Interstate	SC2 : State Contractor
CT2 : County	INT : Interstate	SC3 : State Contractor
CT3 : County	LC1 : Local	SC4 : State Contractor
CT4 : County	LC2 : Local	SC6 : State Contractor
CT6 : County	LC3 : Local	SO1 : Other - State
EC1 : EPA Contractor	LC4 : Local	SO2 : Other - State
EC2 : EPA Contractor	LC6 : Local	SO3 : Other - State
EC3 : EPA Contractor	LCL : Local	SO4 : Other - State
EC4 : EPA Contractor	MN1 : Municipal	SO6 : Other - State
EC6 : EPA Contractor	MN2 : Municipal	ST1 : State
EO1 : Other - EPA	MN3 : Municipal	ST2 : State
EO2 : Other - EPA	MN4 : Municipal	ST3 : State

Agency Type Code Listing		
EO3 : Other - EPA	MN6 : Municipal	ST4 : State
EO4 : Other - EPA	MUN : Municipal	ST6 : State
EO6 : Other - EPA	OF1 : Other Federal	STA : State
EP1 : U.S. EPA	OF2 : Other Federal	STC : State Contractor
EP2 : U.S. EPA	OF3 : Other Federal	STF : State - Using
EP3 : U.S. EPA	OF4 : Other Federal	Federal Credentials
EP4 : U.S. EPA	OF6 : Other Federal	STO : Other - State
EP6 : U.S. EPA	OFD : Other Federal	TR1 : Tribal
EPA : U.S. EPA	REG : Regional	TR2 : Tribal
EPC : EPA Contractor	RG1 : Regional	TR3 : Tribal
EPO : Other - EPA	RG2 : Regional	TR4 : Tribal
IN1 : Interstate	RG3 : Regional	TR6 : Tribal
IN2 : Interstate	RG4 : Regional	TRB : Tribal
IN3 : Interstate	RG6 : Regional	TRF : Tribal - Using Federal Credentials

— Click *Save and Add Another*.
— Repeat these steps until all the desired Agency Maps have been added to the table.

Customize the Subscriber Agreement

5. On the ***Create Instance*** page, customize the subscriber agreement by providing information on the Regulatory Authority and entering any specific terms and conditions

— Provide the mailing information for the Regulatory Authority.
  - Enter name of the Regulatory Authority.
  - Enter the address.
  - Enter the supplemental address.
  - Enter the City.
  - Select the State.
  - Enter the Zip Code.

— Enter the opening terms of the subscriber agreement.
— Enter the closing terms of the subscriber agreement.
  - Enter a condition for the subscriber agreement.
  - Click Save and Add Another.
  - Click on the move up arrow or move down arrow to change the order of the conditions.
  - Repeat this step until all the required conditions are added to the table.

— Click Save.

Create the Initial Internal Administrator

6. On the ***Create Instance*** page, provide the account information for the first Internal Administrator for this instance:
   — Enter the Internal Administrator's email address.

- Confirm the Internal Administrator's email address by entering it a second time.
- Choose whether to use the Internal Administrator's email address as the user name or create a custom user name.
- If you choose to create a custom user name, enter the user name.
- Enter the Internal Administrator's first name.
- Enter the Internal Administrator's last name.
- Enter the Internal Administrator's 10-digit telephone number in the following format (xxx-xxx-xxxx).
- Enter the name of the organization with which the Internal Administrator is affiliated.
- Enter a temporary password
- Confirm the temporary password by typing it again.

Save the Instance

7. Review the information on the ***Confirm Create Instance*** page and click *Create*.

You will be returned to your ***Home*** page and the newly created instance will be displayed in the ***My Instances*** table.

8.0 Compiling and Deploying NetDMR from Source Code

Follow the steps in this section to build and deploy NetDMR from source:

8.1 Step 1: Environment

To build NetDMR from source, you must have the following applications installed on your computer.

- Maven 2 (e.g., 2.0.4)
- JDK 1.5 (e.g., Sun JDK)

8.2 Step 2: Source

The source code for NetDMR is available in as follows:

- Exchange Network Site for NetDMR – The NetDMR source code and associated documentation has been posted to the following Exchange Network for download:

  http://www.exchangenetwork.net/exchanges/water/netdmr.htm.

- Subversion Repository – ERG utilized the source code control application Subversion to maintain a NetDMR source and a history of changes. The Subversion Repository can be made available upon request.

8.3 Step 3: Properties

Each deployment environment requires specific customizations for the environment. These are specified through a property file at runtime. The build system injects the property file with values from filters found in the filters directory netdmr-web/src/main/filters.

When performing a build, two property files are used based on the environment requested to be built. One file contains the general properties used in the application, and the other contains the database specific information. This allows the same general properties to be combined with a database specific property file at build time. Maven profiles are used to specify which property files will be used. For example, to build against oracle you would specify the profiles local,oracle (e.g. mvn package –Plocal,oracle). The src distribution contains the property files for the local environment. These should be changed to reflect the target deployment environment.

- local.properties
- local-db-oracle.properties

Refer to the Section 6.5 for additional detail on the properties.

8.4 Step 4: Database Resource

NetDMR requires a database JNDI resource to be made available in the J2EE container (e.g. JBoss) for the chosen database platform (e.g. oracle or postgres). The JNDI name chosen for the resource must be provided in the NetDMR properties file as specified in the Properties section. The JBoss server will also need an appropriate JDBC driver for the chosen database platform (e.g., ojdbc14-10.2.0.2.0.jar for oracle).

8.5 Step 5: Specify Maven Repository

The netdmr/pom.xml file includes an ERG-specific Maven repository. You must update the repository to match one appropriate for your organization. The jar files required for NetDMR to compile can be determined by viewing the dependencies in each of the pom files, or using Maven to provide a listing.

Note that the NetDMR source package includes only those custom jar files developed during NetDMR implementation. The jar files for the maven repositories are all freely available and can typically be downloaded from the centralized Maven repository (see http://maven.apache.org/ for more information). If any artifact is not available in the central Maven repository then the url will be provided in the error message when you try to build.

A simplified approach to determining files missing from your repository is to attempt to build the application and review Maven outputs that indicate which jar files are missing from your repository.

Some users have reported that Maven was unable to automatically download two of the required dependencies. If this problem occurs on your system you will need to download them and manually add them to your maven repository. These two dependencies and their reference URL are shown below:

- jta-1.0.1B.jar => http://download.java.net/maven/2/javax/transaction/jta/1.0.1B/
- trove-2.0.3.jar => http://sourceforge.net/project/showfiles.php?group_id=39235

8.6 Step 6: Build/Deploy NetDMR Application

To build NetDMR, complete the following.

1. Obtain the source code for NetDMR either through the Exchange Network Site or ERG's Subversion repository (and unzip).
2. Open a command window with the path set to the top folder of the NetDMR source code.
3. Type 'mvn install –Plocal,oracle,nowardeploy'. This will build the application. The war file 'netdmr-web-1.0-SNAPSHOT.war' will be generated in the netdmr-web/target directory.
4. Rename the war file to the name you would like NetDMR to be deployed under (e.g. netdmr-web-pr-1) in your environment
5. Copy the war file to the appropriate JBoss server.

9.0 Monitoring and Maintaining NetDMR

After NetDMR is fully operational, several routine tasks which are recommended to monitor the application to assure smooth operation.

9.1 Database and Server Monitoring

NetDMR is intended to handle a large amount of data during operation; routine database maintenance will be critical to the overall performance, reliability, and security of the application. The following tasks are highly recommended on a routine basis:

- ***Analyze Tables***: The tables in the database should be analyzed on a routine basis in order to maintain accurate and up-to-date statistics. These statistics are used by the database software to determine the optimum query execution path and data retrieval. Outdated statistics will cause the database software to choose a sub-optimal execution plan and will degrade the database performance. A minimum base line for analyzing tables would be to analyze all tables once a week. A more advanced routine would be to analyze a table only after the data in the table had changed by more than 30 percent.

- ***PostgreSQL Vacuum Tables***: When data is deleted in the PostgreSQL environment it is actually only marked for deletion and is not truly removed until a VACUUM is performed on the table. It is recommended that a VACUUM be performed on all tables on a nightly basis. Note that a VACUUM can be performed while the database is online. A more powerful version of the VACUUM command is "VACUUM FULL" which will not only remove the deleted records but also compact the table to better organize the remaining data. This form of the command can only be run when the database is off line and it is recommended to be run once a week.

- ***Backups:*** As with any database, it is critical that backups be maintained of the database to guard against data loss. Your existing server environment should already contain a robust backup system and NetDMR should be included in that system before it is made available for production.

- ***Storage Space:*** By default, NetDMR keeps all CORs for a minimum of 6 years, along with all associated supporting data, user accounts, and authentication logs. Due to this, the size of the NetDMR database will increase steadily. The disk usage of the database should be monitored and the available space should be increased when needed to provide enough storage space.

- ***Memory Usage***: The memory usage of NetDMR will be dependent on the number of users logged in, the processing of the transactions with the

Node, and any DMR imports completed by users. As with any application, the more activities overlap, the higher the memory usage will be. The typical memory usage should be tracked, reviewed periodically, and expanded as necessary.

9.2 Log File Monitoring

The amount of logging performed by NetDMR can be adjusted to one of five levels: debug, info, warn, error, and fatal. The debug level is the finest grain and should only be used during development or testing. It is recommended that a production application be deployed with either warn, error, or fatal.

After deploying NetDMR in a typical JBoss server environment, it will log all events to a log file in real time, with a new file created at the start of each day. It is recommended that the log files be reviewed periodically for any error messages. Note that if the log file is one or two orders of magnitude larger than typical, this usually indicates a communication failure between the JBoss server and database server.

9.3 Update Installation Settings

Complete the steps to view and edit installation settings:

1. Login to NetDMR as a System Administrator.
2. Click **Setting** under the **Manage** menu on the ***Home*** page.
3. Click *Edit Settings* on the ***View Settings*** page.
4. Click in the box next to User/COR Purge Schedule to change the setting.
5. Click in the box next to Time Zone to change the setting.
6. Click in the box next to Suspect Analysis Frequency to change the setting.
7. Click in the box next to Suspect Analysis Log Data Subset to change the setting.
8. In the NetDMR Signing Certificate table, click in the Select Certificate column to specify the active signing certificate.
9. Click *Save*.
10. Click *Save* again on the ***Confirm Edit Settings*** page to finalize your modifications.

9.4 Create a New Instance

An instance is a customized version of NetDMR, specific to a Regulatory Authority. Follow these directions to create a new instance in NetDMR and specify general information, create the initial internal administrator, assign Agency Maps, and update the subscriber agreement.

Create General Information

1. Log into NetDMR with the System Administrator account.
2. Click Create New Instance to access the ***Create Instance*** page.
3. On the ***Create Instance*** page:
   - — Enter the name of the instance.
   - — Select the time zone.

- Select the number of security questions users must answer when creating an account.
- Select how often users will be required to change their password.
- Select whether to allow multiple signatory requests in a subscriber agreement.
- Upload the graphic that will be shown in the top right corner of each page by clicking *Browse* and selecting it from your hard drive. Graphics must be in the *.jpg format. The recommended size specifications are:
  - Height: 60 pixels
  - Width: 20 pixels

  Graphics of a different size will likely appear distorted.
- Specify the Submittal Postal Code. The DMR data flow requires that NetDMR include in the submission file name a submittal Postal Code. NetDMR uses the code you provide to properly name all DMR submission files that are submitted to CDX and ICIS. You will need to coordinate with EPA ICIS staff on the proper postal code for each instance you create.
- Specify a Contact Name. This name will be displayed on the NetDMR Contact Information page.
- Specify a Contact Email. This email will be displayed on the NetDMR Contact Information page and the Home page.
- Specify a Contact Phone. This phone number will be displayed on the NetDMR Contact Information page.

Specify Agency Maps

4. On the ***Create Instance*** page, specify the Agency Maps by selecting the states or areas appropriate for this instance. You should consult with the program area or business lead for the Instance to determine appropriate Agency Maps.
   - Select the permit prefix.
   - Select the agency type code. This is the code used in ICIS-NPDES to categorize the agency that administers the permit. Options, as of August 2008, are as follows:

Agency Type Code Listing		
CNT : County	IN4 : Interstate	SC1 : State Contractor
CT1 : County	IN6 : Interstate	SC2 : State Contractor
CT2 : County	INT : Interstate	SC3 : State Contractor
CT3 : County	LC1 : Local	SC4 : State Contractor
CT4 : County	LC2 : Local	SC6 : State Contractor
CT6 : County	LC3 : Local	SO1 : Other - State
EC1 : EPA Contractor	LC4 : Local	SO2 : Other - State
EC2 : EPA Contractor	LC6 : Local	SO3 : Other - State
EC3 : EPA Contractor	LCL : Local	SO4 : Other - State
EC4 : EPA Contractor	MN1 : Municipal	SO6 : Other - State
EC6 : EPA Contractor	MN2 : Municipal	ST1 : State
EO1 : Other - EPA	MN3 : Municipal	ST2 : State
EO2 : Other - EPA	MN4 : Municipal	ST3 : State
EO3 : Other - EPA	MN6 : Municipal	ST4 : State

Agency Type Code Listing		
EO4 : Other - EPA	MUN : Municipal	ST6 : State
EO6 : Other - EPA	OF1 : Other Federal	STA : State
EP1 : U.S. EPA	OF2 : Other Federal	STC : State Contractor
EP2 : U.S. EPA	OF3 : Other Federal	STF : State - Using Federal Credentials
EP3 : U.S. EPA	OF4 : Other Federal	STO : Other - State
EP4 : U.S. EPA	OF6 : Other Federal	TR1 : Tribal
EP6 : U.S. EPA	OFD : Other Federal	TR2 : Tribal
EPA : U.S. EPA	REG : Regional	TR3 : Tribal
EPC : EPA Contractor	RG1 : Regional	TR4 : Tribal
EPO : Other - EPA	RG2 : Regional	TR6 : Tribal
IN1 : Interstate	RG3 : Regional	TRB : Tribal
IN2 : Interstate	RG4 : Regional	TRF : Tribal - Using Federal Credentials
IN3 : Interstate	RG6 : Regional	

— Click *Save and Add Another*.
— Repeat these steps until all the desired Agency Maps have been added to the table.

Customize the Subscriber Agreement

5. On the ***Create Instance*** page, customize the subscriber agreement by providing information on the Regulatory Authority and entering any specific terms and conditions

— Provide the mailing information for the Regulatory Authority.
  - Enter name of the Regulatory Authority.
  - Enter the address.
  - Enter the supplemental address.
  - Enter the City.
  - Select the State.
  - Enter the Zip Code.

— Enter the opening terms of the subscriber agreement.
— Enter the closing terms of the subscriber agreement.
  - Enter a condition for the subscriber agreement.
  - Click Save and Add Another.
  - Click on the move up arrow or move down arrow to change the order of the conditions.
  - Repeat this step until all the required conditions are added to the table.

— Click *Save*.

Create the Initial Internal Administrator

6. On the ***Create Instance*** page, provide the account information for the first Internal Administrator for this instance:
   — Enter the Internal Administrator's email address.

- — Confirm the Internal Administrator's email address by entering it a second time.
- — Choose whether to use the Internal Administrator's email address as the user name or create a custom user name.
- — If you choose to create a custom user name, enter the user name.
- — Enter the Internal Administrator's first name.
- — Enter the Internal Administrator's last name.
- — Enter the Internal Administrator's 10-digit telephone number in the following format (xxx-xxx-xxxx).
- — Enter the name of the organization with which the Internal Administrator is affiliated.
- — Enter a temporary password
- — Confirm the temporary password by typing it again.

Save the Instance

7. Review the information on the ***Confirm Create Instance*** page and click *Create*.

You will be returned to your ***Home*** page and the newly created instance will be displayed in the ***My Instances*** table.

9.5 Manage Instances

The following section provides information on how to manage your instances. These topics include.

- View Instance Status
- Edit Instance Settings
- Delete Instance

9.5.1 View Instance Status

You can access the details of your instances by logging into your account and clicking on the name of the instance in the ***My Instances*** table on your ***Home*** page. Here you will be able to view the following information on each of your instances:

- Instance Details
- Internal Administrators
- Agency Maps
- Subscriber Agreement

Note: The ***My Instances*** table is also accessible by clicking on **Instances** under the **Manage** menu.

Instance Details

The Instance Details section displays the information that you specified when you created this instance, including:

- Instance Name
- Instance Time Zone
- Number of Secret Questions
- Password Change Frequency
- Multiple requests per subscriber agreement?
    - Instance Status
- Submittal Postal Code
- Contact Name
- Contact Email
- Contact Phone

Internal Administrators

The Internal Administrators table displays the User Name, Affiliation, Email Address, and Phone Number of each Internal Administrator. Up to ten Internal Administrators will be displayed by default. If more than ten Internal Administrators are associated with your instance, you can navigate through the list by:

- Clicking the green forward arrow to display the next page of Internal Administrators.
- Clicking the green back arrow to view the previous page of Internal Administrators.
- Clicking the double green forward arrow icon to display the last ten Internal Administrators.
- Clicking the double green back arrow icon to display the first ten Internal Administrators.
- Clicking the View All link to show all Internal Administrators in the table on the same page with the default sort applied.
- Clicking the View Partial link to return from the View All display back to viewing ten Internal Administrators at a time with the default sort applied.

You can click any underlined column title to sort the Internal Administrators in ascending or descending order by the information in that column.

Agency Maps

The Agency Maps table displays the selected Permit Prefixes and associated Agency Type Codes. Up to ten Agency Maps will be displayed by default. If more than ten Agency Maps are associated with your instance, you can navigate through the list by:

- Clicking the green forward arrow to display the next page of Agency Maps.
- Clicking the green back arrow to view the previous page of Agency Maps.
- Clicking the double green forward arrow icon to display the last ten Agency Maps.
- Clicking the double green back arrow icon to display the first ten Agency Maps.

- Clicking the View All link to show all Agency Maps in the table on the same page with the default sort applied.
- Clicking the View Partial link to return from the View All display back to viewing 10 Agency Maps at a time with the default sort applied.

You can click the Permit Prefix or Agency type code column title to sort the Agency Maps in ascending or descending order.

Subscriber Agreement

The Subscriber Agreement section displays the following information:

- Regulatory Authority Mailing Information
  - Attn
  - Address
  - Supplemental Address
  - City
  - State
  - Zip Code
- Terms and Conditions
  - Opening Terms
  - Closing Terms
  - A table displaying each condition

9.5.2 Edit Instance

You can edit instances by logging into your account and clicking on the name of the instance in the ***My Instances*** table on your ***Home*** page, then click the *Edit Instance* button. You can then modify the following information from the ***Edit Instance*** page that is displayed:

- Instance Details
- Internal Administrators
- Agency Maps
- Subscriber Agreement

Instance Details

You can edit the following information in the Instance Details section, including:

- Instance Name
- Instance Time Zone
- Number of Secret Questions
- Password Change Frequency
- Multiple requests per subscriber agreement?
  - Instance Status
- Submittal Postal Code
- Contact Name

- Contact Email
- Contact Phone

Click Save on the ***Edit Instance*** page and then click Save on the ***Confirm Edit Instance*** page to finalize your changes.

Internal Administrators

You can update a view only Internal User to an Internal Administrator role by clicking in the box next to Internal Administrator, selecting the name of the user to grant Administrator access to, and click *Save and Add Another*.

You can also delete Internal Administrator access from a user by clicking the icon in the Delete column in the row for the user of interest.

To complete your changes, click *Save* on the ***Edit Instance*** page and then click *Save* on the ***Confirm Edit Instance*** page.

The Internal Administrators table displays the User Name, Affiliation, Email Address, and Phone Number of each Internal Administrator. Up to ten Internal Administrators will be displayed by default. If more than ten Internal Administrators are associated with your instance, you can navigate through the list by:

- Clicking the green forward arrow to display the next page of Internal Administrators.
- Clicking the green back arrow to view the previous page of Internal Administrators.
- Clicking the double green forward arrow icon to display the last ten Internal Administrators.
- Clicking the double green back arrow icon to display the first ten Internal Administrators.
- Clicking the View All link to show all Internal Administrators in the table on the same page.
- Clicking the View Partial link to return from the View All display back to viewing ten Internal Administrators at a time.

You can click any underlined column title to sort the Internal Administrators in ascending or descending order by the information in that column.

Agency Maps

You can edit the Agency Maps for the instance in the Agency Maps section.

- Delete an Agency Map: Click the icon in the delete column for the row of interest and click OK to confirm the deletion.
- Update an Agency Map: Click the pencil icon in the update column for the row of interest, click to select a new Agency Type code, and click Save to confirm the selection.

- Add new Agency Maps:
  - Select the permit prefix.
  - Select the agency type code.
  - Click Save and Add Another.
  - Repeat these steps until all the desired Agency Maps have been added to the table.
- Click *Save* on the ***Edit Instance*** page and then click Save on the ***Confirm Edit Instance*** page to finalize your changes.

Up to ten Agency Maps will be displayed by default. If more than ten Agency Maps are associated with your instance, you can navigate through the list by:

- Clicking the green forward arrow to display the next page of Agency Maps.
- Clicking the green back arrow to view the previous page of Agency Maps.
- Clicking the double green forward arrow icon to display the last ten Agency Maps.
- Clicking the double green back arrow icon to display the first ten Agency Maps.
- Clicking the View All link to show all Agency Maps in the table on the same page.
- Clicking the View Partial link to return from the View All display back to viewing ten Agency Maps at a time.

You can click the Permit Prefix or Agency type code column title to sort the Agency Maps in ascending or descending order.

Subscriber Agreement

You can edit any of the following information for the Subscriber Agreement:

- Regulatory Authority Mailing Information
  - Attn
  - Address
  - Supplemental Address
  - City
  - State
  - Zip Code
- Terms and Conditions
  - Opening Terms
  - Closing Terms
- Add a new condition by entering the condition text, click *Save and Add Another*.
  - Click on the move up arrow or move down arrow to change the order of the conditions
  - Repeat this step until all the required conditions are added to the table.

- To delete a condition, click on the icon in the row of interest and click *OK* to confirm the deletion.

Click *Save* on the ***Edit Instance*** page and then click *Save* on the ***Confirm Edit Instance*** page to finalize your changes.

9.5.3 Delete Instance

You can delete an instance by logging into your account, and clicking the icon in the Delete column for the instance of interest on the ***My Instances*** page. Click *Delete* on the ***Confirm Delete Instance*** page to finalize the deletion. **If you delete an instance, all information associated with the instance, including users, DMRs, and CORs, will be removed and will no longer be available.**

9.6 Manage Downtime

You can schedule downtimes for any of the instances in your installation. During a downtime, external users can not access the Instance, internal users with a view role cannot access the Instance, and new user accounts cannot be created. Note that Internal Administrators can still access an Instance during a downtime. To schedule a downtime for the entire NetDMR installation or one or more instances, follow the instructions below:

1. Click **Downtimes** under the **Manage** menu on the ***Home*** page.
2. Click *Schedule Another Downtime* on the ***Instance Downtime Schedule*** page
3. Click in the box next to Take Down on the ***Schedule Downtime*** page to select all instances, or one or more specific instances.
4. Enter a Start Date and Start Time. Note that if the installation and instance time zones are different, the downtime start time must be entered in the instance timezone. For example, if your installation timezone is Eastern but the instance timezone is Central, starting a downtime at 2 pm Central would require that you set the downtime start time at 3 pm Eastern.
5. Enter and End Date and End Time Note that if the installation and instance time zones are different, the downtime end time must be entered in the instance timezone. For example, if your installation timezone is Eastern but the instance timezone is Central, ending a downtime at 3 pm Central would require that you set the downtime start time at 4 pm Eastern.
6. Enter a message to be displayed to users.
7. Click *Save* to schedule the downtime.
8. Click *Save* again on the ***Schedule Downtime Confirm*** page.
9. Your downtime will be displayed in the table on the ***Instance Downtime Schedule*** page.

To update a downtime, click the in the row of interest on the Schedule Downtime table, update the downtime settings, and click *Save*.

To delete a downtime, click the in the row of interest on the Schedule Downtime table, and click *OK* to verify the deletion. Note that you cannot delete a downtime that has already started.

9.7 Reference Table Management

NetDMR includes a copy of required ICIS-NPDES reference tables, current as of August 31, 2008. A single set of reference tables is used across an entire NetDMR installation (e.g., EPA installation). All instances (e.g., Region1, Region2, Rhode Island) within a single installation use the same set of reference tables. For example, the same list of Agency Type Codes is used by all instances. The ICIS-NPDES reference tables that are used by NetDMR include:

- REF_AGENCY_TYPE
- REF_FREQUENCY_OF_ANALYSIS
- REF_MONITORING_LOCATION
- REF_NODI
- REF_PARAMETER
- REF_PERMIT_STATUS
- REF_PERM_FEATURE_TYPE
- REF_SAMPLE_TYPE
- REF_STATISTICAL_BASE
- REF_UNIT
- REF_UNIT_GROUP
- XREF_UNIT_GROUP_UNIT

The reference tables are self explanatory, with the exception of the unit and unit group related reference codes. A NetDMR user can select a different unit code than that specified for the parameter. However, the list of available units that the user can select depends on the parameter unit group. The REF_PARAMETER table specifies the unit group. The XREF_UNIT_GROUP_UNIT relates unit groups to the available units within that group.

The reference tables will be similar to those stored in ICIS-NPDES to simplify maintenance. Each type of reference code (e.g., Agency Type) will be stored in a distinct table (e.g., ref_agency_type). The name of each reference table will be prefixed with 'ref_' for easy identification. Each table will also contain a *Code*, *Description,* and *Status* column similar to the ICIS-NPDES counterpart. An ID column will be included to provide a constant unique key. In general, each reference table will include the following:

- ID: A unique key for each option. The ID is generated by NetDMR and is used by other tables within the NetDMR database to link to one of the reference codes. An ID will never be changed once assigned.

- Code: The code is a short abbreviation that uniquely identifies the option. The code is assigned outside of NetDMR and is used to link the codes between systems (e.g., NetDMR and ICIS-NPDES). The code is used in the various Exchange Network data flows to identify which of the available options was selected. If a NetDMR user has the option to select one or more of the options, the user is presented with the list of codes from which to choose. The code must be unique within the reference table.

- Description: The description contains a human readable description of the option. If a NetDMR user has the option to select one or more of the options, the user can view a list of the descriptions for each of the codes.

- Status: The status column indicates whether the code is Active (A) or Inactive (I). An active code should be displayed in the list of available options when displayed to a user. An inactive code should not appear in such a list, but needs to be maintained for historical purposes since records in other tables within the database may reference the code.

NetDMR supports only manual updates to the reference tables. An authorized person (e.g., database administrator) must write insert, update, and delete SQL statements as necessary to update the reference tables.

Adding a New Option

Adding a new code requires inserting a new record in to the appropriate reference table; no other changes are required. The NetDMR database automatically generates an *ID* for the new row. An example insert statement for adding a new code is as follows:

```
INSERT INTO ref_agency_type (code, description, status) VALUES
('EXP', 'Example Code', 'A')
```

Updating an Existing Option

The Code, Description, and Status columns are the only reference table columns that should be updated. The ID column is auto-generated and should never be updated since NetDMR uses this column within other NetDMR tables to link to a particular option in the reference table. The following is an example UPDATE statement to update the description for one of the options.

```
UPDATE ref_agency_type SET description='Revised Example Code'
WHERE code='EXP'
```

Deleting an Option

An option in the reference table should only be deleted when the option is no longer used anywhere in the NetDMR database, and will never be used in an Exchange Network data flow. Even in these cases, it may be preferable to update the option to a status code of 'I' (Inactive) rather than deleting the option. The following is an example DELETE statement.

```
DELETE FROM ref_agency_type WHERE ID = 9
```

10.0 NetDMR Dependencies

NetDMR top level, business, persistence, and web dependencies are provided in this section.

10.1 NetDMR Top-level Dependencies

10.1.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
commons-discovery	commons-discovery	0.4	-	jar	
org.acegisecurity	acegi-security	1.0.6	-	jar	
org.apache.axis	axis	1.4	-	jar	
org.hibernate	hibernate	3.2.5.ga	-	jar	
org.springframework	spring	2.5.4	-	jar	

10.1.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.mail	mail	1.4	-	jar	
org.easymock	easymock	2.3	-	jar	
org.springframework	spring-test	2.5.4	-	jar	

10.1.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.sun.xml.rpc	jaxrpc-spi	1.1.3_01	-	jar	
javax.servlet	servlet-api	2.4	-	jar	
javax.xml.soap	saaj-api	1.3	-	jar	

10.1.4 Project Transitive Dependencies

The following is a list of transitive dependencies for this project. Transitive dependencies are the dependencies of the project dependencies.

10.1.4.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
antlr	antlr	2.7.6	-	jar	
asm	asm	1.5.3	-	jar	
asm	asm-attrs	1.5.3	-	jar	
cglib	cglib	2.1_3	-	jar	
commons-codec	commons-codec	1.3	-	jar	
commons-collections	commons-collections	3.1	-	jar	
commons-lang	commons-lang	2.1	-	jar	
commons-logging	commons-logging	1.0.4	-	jar	
dom4j	dom4j	1.6.1	-	jar	
javax.transaction	jta	1.0.1B	-	jar	
log4j	log4j	1.2.13	-	jar	
net.sf.ehcache	ehcache	1.2.3	-	jar	
oro	oro	2.0.8	-	jar	

10.1.4.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
junit	junit	4.4	-	jar	

10.1.4.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.activation	activation	1.1	-	jar	
javax.xml	jaxrpc-api	1.1	-	jar	

10.1.5 Project Dependency Graph

10.1.5.1 Dependency Tree

- net.exchangenetwork.netdmr:netdmr-api:jar
  - com.sun.xml.rpc:jaxrpc-spi:jar
    - javax.xml:jaxrpc-api:jar
  - commons-discovery:commons-discovery:jar
  - javax.mail:mail:jar
  - org.apache.axis:axis:jar

- javax.servlet:servlet-api:jar
- org.acegisecurity:acegi-security:jar
  - commons-lang:commons-lang:jar
  - commons-codec:commons-codec:jar
  - oro:oro:jar
  - log4j:log4j:jar
- org.easymock:easymock:jar
- org.hibernate:hibernate:jar
  - net.sf.ehcache:ehcache:jar
  - javax.transaction:jta:jar
  - asm:asm-attrs:jar
  - dom4j:dom4j:jar
  - antlr:antlr:jar
  - cglib:cglib:jar
  - asm:asm:jar
  - commons-collections:commons-collections:jar
- org.springframework:spring-test:jar
  - junit:junit:jar
- org.springframework:spring:jar
  - commons-logging:commons-logging:jar
- javax.xml.soap:saaj-api:jar
  - javax.activation:activation:jar

10.1.5.2 Dependency Listings

- NetDMR Application :: APIs

NetDMR Appication Tier APIs
http://www2.ergweb.com/projects/netdmr/netdmr-api

- Unnamed - com.sun.xml.rpc:jaxrpc-spi:jar:1.1.3_01

- Java API for XML Based RPC

Part of the Java Web Services Developer Pack 1.6
http://java.sun.com/webservices/jaxrpc/index.jsp

- Discovery

Commons Discovery
http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- JavaMail API

The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications.
https://glassfish.dev.java.net/javaee5/mail/

- Unnamed - org.apache.axis:axis:jar:1.4

- POM was created from deploy:deploy-file

- Unnamed - javax.servlet:servlet-api:jar:2.4

- Acegi Security Core

Acegi Security System for Spring
http://acegisecurity.org/acegi-security

- Lang

Commons.Lang, a package of Java utility classes for the classes that are in java.lang's hierarchy, or are considered to be so standard as to justify existence in java.lang.
http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- Codec

The codec package contains simple encoder and decoders for various formats such as Base64 and Hexadecimal. In addition to these widely used encoders and decoders, the codec package also maintains a collection of phonetic encoding utilities.
http://jakarta.apache.org/commons/codec/

- Unnamed - oro:oro:jar:2.0.8

- Log4j

Log4j
http://logging.apache.org/log4j/docs/

- EasyMock

EasyMock provides Mock Objects for interfaces in JUnit tests by generating them on the fly using Java's proxy mechanism
http://www.easymock.org

- Hibernate

Relational Persistence for Java
http://www.hibernate.org

- ehcache

ehcache is a pure Java, in-process cache with the following features: 1. Fast. 2. Simple. 3. Multiple eviction policies: LRU, LFU and FIFO. 4. Caches can be in memory or on disk. 5. Disk Stores can be persistent between VM restarts. 6. Distributed caching using multicast and RMI, with a pluggable API. 7. Cache and CacheManager listeners 8. Supports multiple Caches per CacheManager, and multiple CacheManagers per application. 9. Acts as a pluggable cache for Hibernate 3.1, 3 and 2.1. 10. Small foot print. Both in terms of size and memory requirements. 11. Minimal dependencies apart from J2SE. 12. Fully documented. See the online Documentation and the online JavaDoc. 13. Comprehensive Test Coverage. See the clover test report. 14. Available under the Apache 1.1 license. EHCache's copyright and licensing has been reviewed and approved by the Apache Software Foundation, making EHCache suitable for use in Apache projects. 15. Production tested. EHCache is used on a large and very busy

eCommerce site. 16. Web caching, pull-through caches and other common caching implementations are provided in the ehcache-constructs module.
http://ehcache.sf.net

- Java Transaction API

The javax.transaction package. It is appropriate for inclusion in a classpath, and may be added to a Java 2 installation.
http://java.sun.com/products/jta

- asm-attrs

http://asm.objectweb.org/

- dom4j

dom4j: the flexible XML framework for Java
http://dom4j.org

- AntLR

http://www.antlr.org/

- cglib

http://cglib.sourceforge.net/

- asm

http://asm.objectweb.org/

- Unnamed - commons-collections:commons-collections:jar:3.1

Types that extend and augment the Java Collections Framework.

- Spring Framework: Test

Spring Framework: Test
http://www.springframework.org

- JUnit

JUnit is a regression testing framework written by Erich Gamma and Kent Beck. It is used by the developer who implements unit tests in Java.
http://junit.org

- Spring Framework

Spring Framework
http://www.springframework.org

- Logging

Commons Logging is a thin adapter allowing configurable bridging to other, well known logging systems.
http://jakarta.apache.org/commons/logging/

- SOAP with Attachments API Package

http://java.sun.com/webservices/saaj/index.jsp

- JavaBeans Activation Framework (JAF)

JavaBeans Activation Framework (JAF) is a standard extension to the Java platform that lets you take advantage of standard services to: determine the type of an arbitrary piece of data; encapsulate access to it; discover the operations available on it; and instantiate the appropriate bean to perform the operation(s).
http://java.sun.com/products/javabeans/jaf/index.jsp

10.2 NetDMR Business - Project Dependencies

10.2.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
net.exchangenetwork.netdmr	netdmr-api	1.0-SNAPSHOT	-	jar	
net.exchangenetwork.netdmr	netdmr-persistence	1.0-SNAPSHOT	-	jar	
net.sf.opencsv	opencsv	1.8	-	jar	
org.codehaus.woodstox	wstx-asl	3.2.6	-	jar	
org.springframework	spring	2.5.4	-	jar	

10.2.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
org.easymock	easymock	2.3	-	jar	
org.springframework	spring-test	2.5.4	-	jar	

10.2.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.sun.xml.rpc	jaxrpc-spi	1.1.3_01	-	jar	
javax.mail	mail	1.4	-	jar	
javax.servlet	servlet-api	2.4	-	jar	
javax.xml.soap	saaj-api	1.3	-	jar	
quartz	quartz	1.5.2	-	jar	
wsdl4j	wsdl4j	1.6.1	-	jar	

10.2.4 Project Transitive Dependencies

The following is a list of transitive dependencies for this project. Transitive dependencies are the dependencies of the project dependencies.

10.2.4.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
antlr	antlr	2.7.6	-	jar	
asm	asm	1.5.3	-	jar	
asm	asm-attrs	1.5.3	-	jar	
cglib	cglib	2.1_3	-	jar	
commons-codec	commons-codec	1.3	-	jar	
commons-collections	commons-collections	3.1	-	jar	
commons-discovery	commons-discovery	0.4	-	jar	
commons-lang	commons-lang	2.1	-	jar	
commons-logging	commons-logging	1.0.4	-	jar	
dom4j	dom4j	1.6.1	-	jar	
javax.transaction	jta	1.0.1B	-	jar	
log4j	log4j	1.2.13	-	jar	
net.sf.ehcache	ehcache	1.2.3	-	jar	
org.acegisecurity	acegi-security	1.0.6	-	jar	
org.apache.axis	axis	1.4	-	jar	
org.hibernate	hibernate	3.2.5.ga	-	jar	
oro	oro	2.0.8	-	jar	
stax	stax-api	1.0.1	-	jar	

10.2.4.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
junit	junit	4.4	-	jar	

10.2.4.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.activation	activation	1.1	-	jar	
javax.xml	jaxrpc-api	1.1	-	jar	

10.2.5 Project Dependency Graph

10.2.5.1 Dependency Tree

- net.exchangenetwork.netdmr:netdmr-business:jar
  - com.sun.xml.rpc:jaxrpc-spi:jar
    - javax.xml:jaxrpc-api:jar
  - net.exchangenetwork.netdmr:netdmr-persistence:jar
  - quartz:quartz:jar
  - javax.mail:mail:jar
  - net.sf.opencsv:opencsv:jar
  - javax.servlet:servlet-api:jar
  - org.easymock:easymock:jar
  - org.springframework:spring-test:jar
    - junit:junit:jar
  - net.exchangenetwork.netdmr:netdmr-api:jar
    - log4j:log4j:jar
    - commons-discovery:commons-discovery:jar
    - org.apache.axis:axis:jar
    - org.acegisecurity:acegi-security:jar
    - org.hibernate:hibernate:jar
  - org.springframework:spring:jar
    - commons-logging:commons-logging:jar
  - org.codehaus.woodstox:wstx-asl:jar
    - stax:stax-api:jar
  - javax.xml.soap:saaj-api:jar
    - javax.activation:activation:jar
  - wsdl4j:wsdl4j:jar

10.2.6 Dependency Listings

- NetDMR Application :: Business Tier

NetDMR Appication Business Tier
http://www2.ergweb.com/projects/netdmr/netdmr-business

- Unnamed - com.sun.xml.rpc:jaxrpc-spi:jar:1.1.3_01

- Java API for XML Based RPC

Part of the Java Web Services Developer Pack 1.6
http://java.sun.com/webservices/jaxrpc/index.jsp

- NetDMR Application :: Persistence Tier

NetDMR Appication Persistence Tier
http://www2.ergweb.com/projects/netdmr/netdmr-persistence

- Unnamed - quartz:quartz:jar:1.5.2

OpenSymphony's Quartz Scheduler

- JavaMail API

The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications.
https://glassfish.dev.java.net/javaee5/mail/

- An open source csv parser for Java.

opencsv is a very simple csv (comma-separated values) parser library for Java. It was developed because all of current csv parsers I've come across don't have commercial-friendly licenses.
http://opencsv.sourceforge.net/

- Unnamed - javax.servlet:servlet-api:jar:2.4

- EasyMock

EasyMock provides Mock Objects for interfaces in JUnit tests by generating them on the fly using Java's proxy mechanism
http://www.easymock.org

- Spring Framework: Test

Spring Framework: Test
http://www.springframework.org

- JUnit

JUnit is a regression testing framework written by Erich Gamma and Kent Beck. It is used by the developer who implements unit tests in Java.
http://junit.org

- NetDMR Application :: APIs

NetDMR Appication Tier APIs
http://www2.ergweb.com/projects/netdmr/netdmr-api

- Log4j

Log4j
http://logging.apache.org/log4j/docs/

- Discovery

Commons Discovery
http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- Unnamed - org.apache.axis:axis:jar:1.4

- POM was created from deploy:deploy-file

- Acegi Security Core

Acegi Security System for Spring
http://acegisecurity.org/acegi-security

- Hibernate

Relational Persistence for Java
http://www.hibernate.org

- Spring Framework

Spring Framework
http://www.springframework.org

- Logging

Commons Logging is a thin adapter allowing configurable bridging to other, well known logging systems.
http://jakarta.apache.org/commons/logging/

- Woodstox

Woodstox is a high-performance XML processor that implements Stax (JSR-173) API
http://woodstox.codehaus.org

- StAX API

StAX API is the standard java XML processing API defined by JSR-173
http://stax.codehaus.org/

- SOAP with Attachments API Package

http://java.sun.com/webservices/saaj/index.jsp

- JavaBeans Activation Framework (JAF)

JavaBeans Activation Framework (JAF) is a standard extension to the Java platform that lets you take advantage of standard services to: determine the type of an arbitrary piece of data; encapsulate access to it; discover the operations available on it; and instantiate the appropriate bean to perform the operation(s).
http://java.sun.com/products/javabeans/jaf/index.jsp

- WSDL4J

Java stub generator for WSDL
http://sf.net/projects/wsdl4j

10.3 NetDMR Persistence Dependencies

10.3.1 compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
net.exchangenetwork.netdmr	netdmr-api	1.0-SNAPSHOT	-	jar	
org.springframework	spring	2.5.4	-	jar	

10.3.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.oracle	ojdbc14	10.2.0.2.0	-	jar	
javax.mail	mail	1.4	-	jar	
org.easymock	easymock	2.3	-	jar	
org.springframework	spring-test	2.5.4	-	jar	
postgresql	postgresql	8.2-504.jdbc3	-	jar	

10.3.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.sun.xml.rpc	jaxrpc-spi	1.1.3_01	-	jar	
javax.servlet	servlet-api	2.4	-	jar	
javax.xml.soap	saaj-api	1.3	-	jar	

10.3.4 Project Transitive Dependencies

The following is a list of transitive dependencies for this project. Transitive dependencies are the dependencies of the project dependencies.

10.3.4.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
antlr	antlr	2.7.6	-	jar	
asm	asm	1.5.3	-	jar	
asm	asm-attrs	1.5.3	-	jar	
cglib	cglib	2.1_3	-	jar	
commons-codec	commons-codec	1.3	-	jar	
commons-collections	commons-collections	3.1	-	jar	
commons-discovery	commons-discovery	0.4	-	jar	
commons-lang	commons-lang	2.1	-	jar	
commons-logging	commons-logging	1.0.4	-	jar	
dom4j	dom4j	1.6.1	-	jar	
javax.transaction	jta	1.0.1B	-	jar	

log4j	log4j	1.2.13	-	jar	
net.sf.ehcache	ehcache	1.2.3	-	jar	
org.acegisecurity	acegi-security	1.0.6	-	jar	
org.apache.axis	axis	1.4	-	jar	
org.hibernate	hibernate	3.2.5.ga	-	jar	
oro	oro	2.0.8	-	jar	

10.3.4.2 test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
junit	junit	4.4	-	jar	

10.3.4.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.activation	activation	1.1	-	jar	
javax.xml	jaxrpc-api	1.1	-	jar	

10.3.5 Project Dependency Graph

10.3.5.1 Dependency Tree

- net.exchangenetwork.netdmr:netdmr-persistence:jar
  - com.sun.xml.rpc:jaxrpc-spi:jar
    - javax.xml:jaxrpc-api:jar
  - javax.mail:mail:jar
  - javax.servlet:servlet-api:jar
  - org.easymock:easymock:jar
  - org.springframework:spring-test:jar
    - junit:junit:jar
  - com.oracle:ojdbc14:jar
  - net.exchangenetwork.netdmr:netdmr-api:jar
    - log4j:log4j:jar
    - commons-discovery:commons-discovery:jar
    - org.apache.axis:axis:jar
    - org.acegisecurity:acegi-security:jar
    - org.hibernate:hibernate:jar
  - postgresql:postgresql:jar
  - org.springframework:spring:jar
    - commons-logging:commons-logging:jar
  - javax.xml.soap:saaj-api:jar
    - javax.activation:activation:jar

10.3.5.2 Dependency Listings

- NetDMR Application :: Persistence Tier

NetDMR Appication Persistence Tier
http://www2.ergweb.com/projects/netdmr/netdmr-persistence

- Unnamed - com.sun.xml.rpc:jaxrpc-spi:jar:1.1.3_01

- Java API for XML Based RPC

Part of the Java Web Services Developer Pack 1.6
http://java.sun.com/webservices/jaxrpc/index.jsp

- JavaMail API

The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications.
https://glassfish.dev.java.net/javaee5/mail/

- Unnamed - javax.servlet:servlet-api:jar:2.4

- EasyMock

EasyMock provides Mock Objects for interfaces in JUnit tests by generating them on the fly using Java's proxy mechanism
http://www.easymock.org

- Spring Framework: Test

Spring Framework: Test
http://www.springframework.org

- JUnit

JUnit is a regression testing framework written by Erich Gamma and Kent Beck. It is used by the developer who implements unit tests in Java.
http://junit.org

- Oracle JDBC Driver

Oracle JDBC driver classes for use with JDK1.4
http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html

- NetDMR Application :: APIs

NetDMR Appication Tier APIs
http://www2.ergweb.com/projects/netdmr/netdmr-api

- Log4j

Log4j
http://logging.apache.org/log4j/docs/

- Discovery

Commons Discovery

http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- Unnamed - org.apache.axis:axis:jar:1.4

- POM was created from deploy:deploy-file

- Acegi Security Core

Acegi Security System for Spring
http://acegisecurity.org/acegi-security

- Hibernate

Relational Persistence for Java
http://www.hibernate.org

- PostgreSQL JDBC Driver

The PostgreSQL JDBC3 Driver for PostgreSQL 8.2 databases
http://jdbc.postgresql.org

- Spring Framework

Spring Framework
http://www.springframework.org

- Logging

Commons Logging is a thin adapter allowing configurable bridging to other, well known logging systems.
http://jakarta.apache.org/commons/logging/

- SOAP with Attachments API Package

http://java.sun.com/webservices/saaj/index.jsp

- JavaBeans Activation Framework (JAF)

JavaBeans Activation Framework (JAF) is a standard extension to the Java platform that lets you take advantage of standard services to: determine the type of an arbitrary piece of data; encapsulate access to it; discover the operations available on it; and instantiate the appropriate bean to perform the operation(s).
http://java.sun.com/products/javabeans/jaf/index.jsp

10.4 NetDMR Web Dependencies

10.4.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.lowagie	itext	1.3	-	jar	
commons-fileupload	commons-fileupload	1.2	-	jar	

commons-io	commons-io	1.2	-	jar	
javax.servlet	jstl	1.1.2	-	jar	
net.exchangenetwork.netdmr	netdmr-api	1.0-SNAPSHOT	-	jar	
net.exchangenetwork.netdmr	netdmr-business	1.0-SNAPSHOT	-	jar	
net.exchangenetwork.netdmr	netdmr-persistence	1.0-SNAPSHOT	-	jar	
opensymphony	sitemesh	2.3	-	jar	
org.springframework	spring-webflow	1.0.5	-	jar	
org.springframework	spring-webmvc	2.5.4	-	jar	
quartz	quartz	1.5.2	-	jar	
struts	struts	1.2.7	-	jar	
taglibs	standard	1.1.2	-	jar	
xerces	xercesImpl	2.4.0	-	jar	
opensymphony	webwork	2.1.5	-	jar	(optional)

10.4.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.mail	mail	1.4	-	jar	
org.easymock	easymock	2.3	-	jar	
org.springframework	spring-test	2.5.4	-	jar	
trove	trove	2.0.3	-	jar	

10.4.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
com.sun.xml.rpc	jaxrpc-spi	1.1.3_01	-	jar	
javax.servlet	servlet-api	2.4	-	jar	
javax.servlet.jsp	jsp-api	2.0	-	jar	
javax.xml.soap	saaj-api	1.3	-	jar	

10.4.4 Project Transitive Dependencies

The following is a list of transitive dependencies for this project. Transitive dependencies are the dependencies of the project dependencies.

10.4.4.1 Compile

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
antlr	antlr	2.7.2	-	jar	
asm	asm	1.5.3	-	jar	
asm	asm-attrs	1.5.3	-	jar	
cglib	cglib	2.1_3	-	jar	
commons-beanutils	commons-beanutils	1.7.0	-	jar	
commons-chain	commons-chain	1.0	-	jar	
commons-codec	commons-codec	1.3	-	jar	
commons-collections	commons-collections	3.1	-	jar	
commons-digester	commons-digester	1.6	-	jar	
commons-discovery	commons-discovery	0.4	-	jar	
commons-lang	commons-lang	2.1	-	jar	
commons-logging	commons-logging	1.0.4	-	jar	
commons-validator	commons-validator	1.1.4	-	jar	
dom4j	dom4j	1.6.1	-	jar	
javax.transaction	jta	1.0.1B	-	jar	
log4j	log4j	1.2.13	-	jar	
net.sf.ehcache	ehcache	1.2.3	-	jar	
net.sf.opencsv	opencsv	1.8	-	jar	
ognl	ognl	2.6.5	-	jar	
opensymphony	oscore	2.2.4	-	jar	
opensymphony	xwork	1.0.3	-	jar	
org.acegisecurity	acegi-security	1.0.6	-	jar	
org.apache.axis	axis	1.4	-	jar	
org.codehaus.woodstox	wstx-asl	3.2.6	-	jar	
org.hibernate	hibernate	3.2.5.ga	-	jar	
org.springframework	spring	2.5.4	-	jar	
org.springframework	spring-binding	1.0.5	-	jar	
oro	oro	2.0.8	-	jar	
stax	stax-api	1.0.1	-	jar	

10.4.4.2 Test

The following is a list of test dependencies for this project. These dependencies are only required to compile and run unit tests for the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
junit	junit	4.4	-	jar	

10.4.4.3 Provided

The following is a list of provided dependencies for this project. These dependencies are required to compile the application, but should be provided by default when using the library:

GroupId	ArtifactId	Version	Classifier	Type	Optional
javax.activation	activation	1.1	-	jar	
javax.xml	jaxrpc-api	1.1	-	jar	

10.4.5 Project Dependency Graph

10.4.5.1 Dependency Tree

- net.exchangenetwork.netdmr:netdmr-web:war
  - net.exchangenetwork.netdmr:netdmr-business:jar
    - net.sf.opencsv:opencsv:jar
    - org.codehaus.woodstox:wstx-asl:jar
      - stax:stax-api:jar
  - net.exchangenetwork.netdmr:netdmr-persistence:jar
  - quartz:quartz:jar
  - javax.mail:mail:jar
  - struts:struts:jar
    - commons-chain:commons-chain:jar
    - commons-digester:commons-digester:jar
      - commons-beanutils:commons-beanutils:jar
    - commons-validator:commons-validator:jar
  - javax.servlet:jstl:jar
  - com.lowagie:itext:jar
  - org.easymock:easymock:jar
  - commons-io:commons-io:jar
  - org.springframework:spring-test:jar
    - junit:junit:jar
  - xerces:xercesImpl:jar
  - opensymphony:webwork:jar
    - opensymphony:oscore:jar
    - opensymphony:xwork:jar
  - commons-fileupload:commons-fileupload:jar
  - taglibs:standard:jar
  - com.sun.xml.rpc:jaxrpc-spi:jar
    - javax.xml:jaxrpc-api:jar
  - trove:trove:jar
  - javax.servlet.jsp:jsp-api:jar
  - org.springframework:spring-webmvc:jar
  - javax.servlet:servlet-api:jar
  - opensymphony:sitemesh:jar
  - net.exchangenetwork.netdmr:netdmr-api:jar
    - log4j:log4j:jar
    - commons-discovery:commons-discovery:jar
    - org.apache.axis:axis:jar
    - org.acegisecurity:acegi-security:jar
    - org.hibernate:hibernate:jar

        - org.springframework:spring:jar
    - org.springframework:spring-webflow:jar
        - org.springframework:spring-binding:jar
            - commons-logging:commons-logging:jar
            - ognl:ognl:jar
    - javax.xml.soap:saaj-api:jar
        - javax.activation:activation:jar

10.4.5.2 Dependency Listings

- NetDMR Application :: Web Application

NetDMR Web
http://www2.ergweb.com/projects/netdmr/netdmr-web

- NetDMR Application :: Business Tier

NetDMR Appication Business Tier
http://www2.ergweb.com/projects/netdmr/netdmr-business

- An open source csv parser for Java.

opencsv is a very simple csv (comma-separated values) parser library for Java. It was developed because all of current csv parsers I've come across don't have commercial-friendly licenses.
http://opencsv.sourceforge.net/

- Woodstox

Woodstox is a high-performance XML processor that implements Stax (JSR-173) API
http://woodstox.codehaus.org

- StAX API

StAX API is the standard java XML processing API defined by JSR-173
http://stax.codehaus.org/

- NetDMR Application :: Persistence Tier

NetDMR Appication Persistence Tier
http://www2.ergweb.com/projects/netdmr/netdmr-persistence

- Unnamed - quartz:quartz:jar:1.5.2

- OpenSymphony's Quartz Scheduler

- JavaMail API

The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications.
https://glassfish.dev.java.net/javaee5/mail/

- Struts

The core of the Struts framework is a flexible control layer based on standard technologies like Java Servlets, JavaBeans, ResourceBundles, and Extensible

Markup Language (XML), as well as various Jakarta Commons packages. Struts encourages application architectures based on the Model 2 approach, a variation of the classic Model-View-Controller (MVC) design paradigm. Struts provides its own Controller component and integrates with other technologies to provide the Model and the View. For the Model, Struts can interact with any standard data access technology, including Enterprise Java Beans, JDBC, and Object Relational Bridge. For the View, Struts works well with JavaServer Pages, including JSTL and JSF, as well as Velocity Templates, XSLT, and other presentation systems. The Struts framework provides the invisible underpinnings every professional web application needs to survive. Struts helps you create an extensible development environment for your application, based on published standards and proven design patterns.
http://struts.apache.org/index.html

- Commons Chain

An implementation of the GoF Chain of Responsibility pattern
http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- Unnamed - commons-digester:commons-digester:jar:1.6

- Unnamed - commons-beanutils:commons-beanutils:jar:1.7.0

- Unnamed - commons-validator:commons-validator:jar:1.1.4
- Unnamed - javax.servlet:jstl:jar:1.1.2

- itext

The iText classes are very useful for people who need to generate read-only, platform independent documents containing text, lists, tables and images. The library is especially useful in combination with Java(TM) technology-based Servlets: The look and feel of HTML is browser dependent; with iText and PDF you can control exactly how your servlet's output will look. iText requires JDK 1.2, and no extra dependencies. It's available for free under a multiple license: MPL and LGPL.
http://www.lowagie.com/iText/

- EasyMock

EasyMock provides Mock Objects for interfaces in JUnit tests by generating them on the fly using Java's proxy mechanism
http://www.easymock.org

- IO

Commons-IO contains utility classes, stream implementations, file filters, and endian classes.
http://jakarta.apache.org/commons/io/

- Spring Framework: Test

Spring Framework: Test

http://www.springframework.org

- JUnit

JUnit is a regression testing framework written by Erich Gamma and Kent Beck. It is used by the developer who implements unit tests in Java.
http://junit.org

- Unnamed - xerces:xercesImpl:jar:2.4.0
- Unnamed - opensymphony:webwork:jar:2.1.5
- Unnamed - opensymphony:oscore:jar:2.2.4
- Unnamed - opensymphony:xwork:jar:1.0.3

- FileUpload

The FileUpload component provides a simple yet flexible means of adding support for multipart file upload functionality to servlets and web applications.
http://jakarta.apache.org/commons/fileupload/

- Unnamed - taglibs:standard:jar:1.1.2
- Unnamed - com.sun.xml.rpc:jaxrpc-spi:jar:1.1.3_01

- Java API for XML Based RPC

Part of the Java Web Services Developer Pack 1.6
http://java.sun.com/webservices/jaxrpc/index.jsp

- Unnamed - trove:trove:pom:2.0.3
- Unnamed - javax.servlet.jsp:jsp-api:jar:2.0

- Spring Framework: Web MVC

Spring Framework: Web MVC
http://www.springframework.org

- Unnamed - javax.servlet:servlet-api:jar:2.4

- Sitemesh

SiteMesh is a web-page layout and decoration framework and web- application integration framework to aid in creating large sites consisting of many pages for which a consistent look/feel, navigation and layout scheme is required.
http://www.opensymphony.com/sitemesh

- NetDMR Application :: APIs

NetDMR Appication Tier APIs
http://www2.ergweb.com/projects/netdmr/netdmr-api

- Log4j

Log4j
http://logging.apache.org/log4j/docs/

- Discovery

Commons Discovery
http://jakarta.apache.org/commons/${pom.artifactId.substring(8)}/

- Unnamed - org.apache.axis:axis:jar:1.4
- POM was created from deploy:deploy-file

- Acegi Security Core

Acegi Security System for Spring
http://acegisecurity.org/acegi-security

- Hibernate

Relational Persistence for Java
http://www.hibernate.org

- Spring Framework

Spring Framework
http://www.springframework.org

- Spring Web Flow

Spring Web Flow
http://www.springframework.org

- Spring Binding

Spring Data Binding Framework
http://www.springframework.org

- Logging

Commons Logging is a thin adapter allowing configurable bridging to other, well known logging systems.
http://jakarta.apache.org/commons/logging/

- Unnamed - ognl:ognl:jar:2.6.5

- SOAP with Attachments API Package

http://java.sun.com/webservices/saaj/index.jsp

- JavaBeans Activation Framework (JAF)

JavaBeans Activation Framework (JAF) is a standard extension to the Java platform that lets you take advantage of standard services to: determine the type of an arbitrary piece of data; encapsulate access to it; discover the operations available on it; and instantiate the appropriate bean to perform the operation(s).
http://java.sun.com/products/javabeans/jaf/index.jsp

11.0 NetDMR Database

Figures 11-1 through 11-4 show the NetDMR database design, including tables, fields, and relationship between the tables that support Common Component and Administrator functionality. The diagram uses the following conventions.

- PK: This represents the primary key for the table. A primary key uniquely identifies a row within a table.
- FK: This represents a foreign key. A foreign key is used to link two tables together.

The tables are grouped according to the overall purpose of the table. A brief description of the each group of tables is provided below. The data dictionary in Appendix F provides detailed information for the database tables and fields including field types, sizes, and comments.

User Information: These tables contain information about the user, including account information, associated logs, and available security questions.

User Permissions: These tables contain information about access control, such as the available permissions, roles, and user types. It also contains which roles each user is assigned.

Instance Settings: These tables contain information for a particular NetDMR instance, such as the number of security questions each user must answer, as well as the instance specific terms and conditions used in Subscriber Agreements.

Permit and DMR Information: These tables contain information about the permits and empty slots. It also contains the information entered by a user when completing a DMR, and the Copy of Record (COR).

Queue for Transactions with the Node: These tables contain information about the communication between NetDMR and an Exchange Network node (e.g., CDX). The tables include requests that were sent and the result that was returned.

Reference Tables: A reference table contains a list of codes and descriptions that are applicable for a data element.

System Settings: These tables contain information about the NetDMR installation and the instances that have been created for the installation.

User Imported DMR: These tables contain information about files a user has uploaded to populate the data within one or more DMRs.

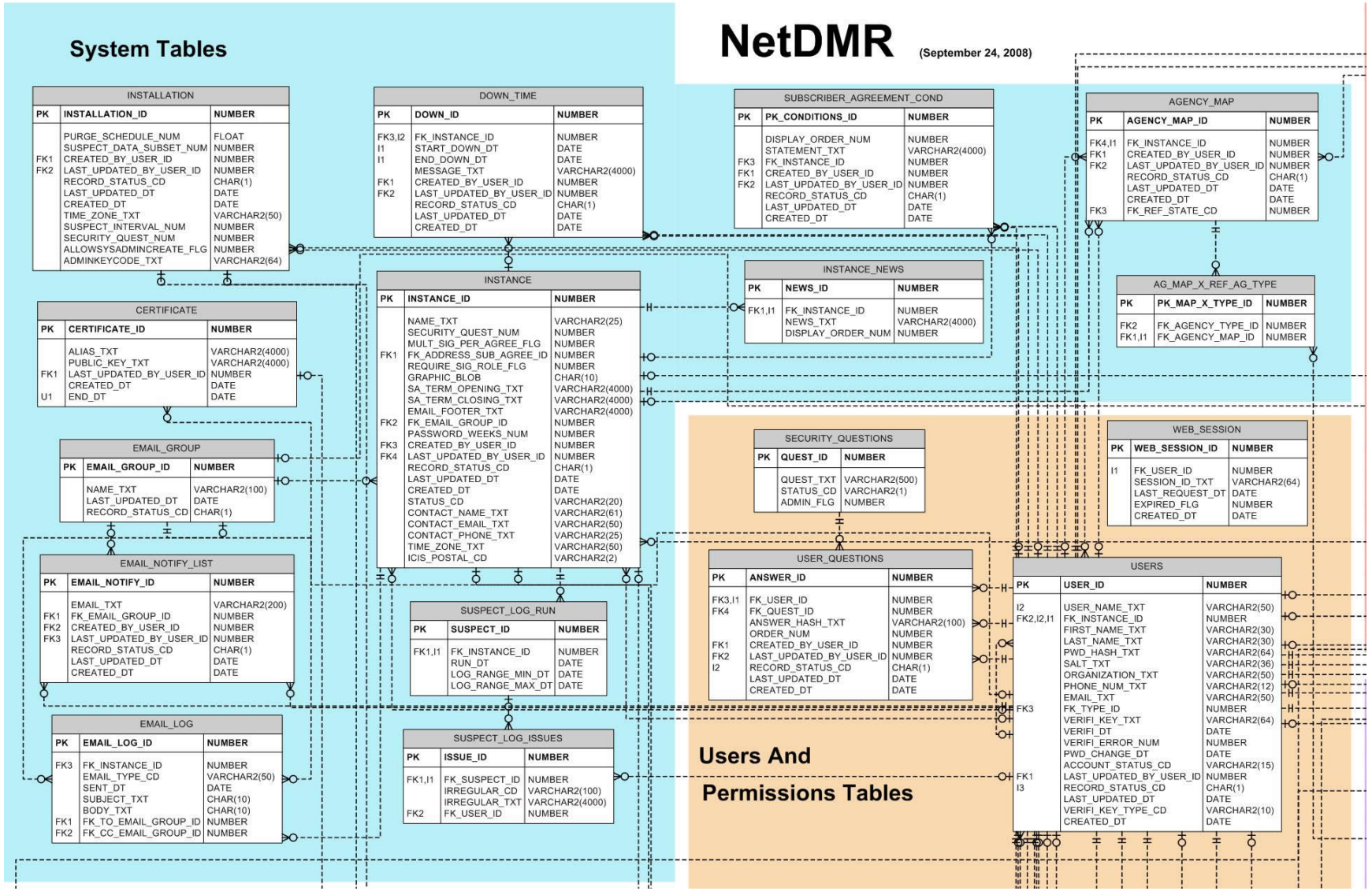

Figure 11-1. NetDMR Database Design Part 1

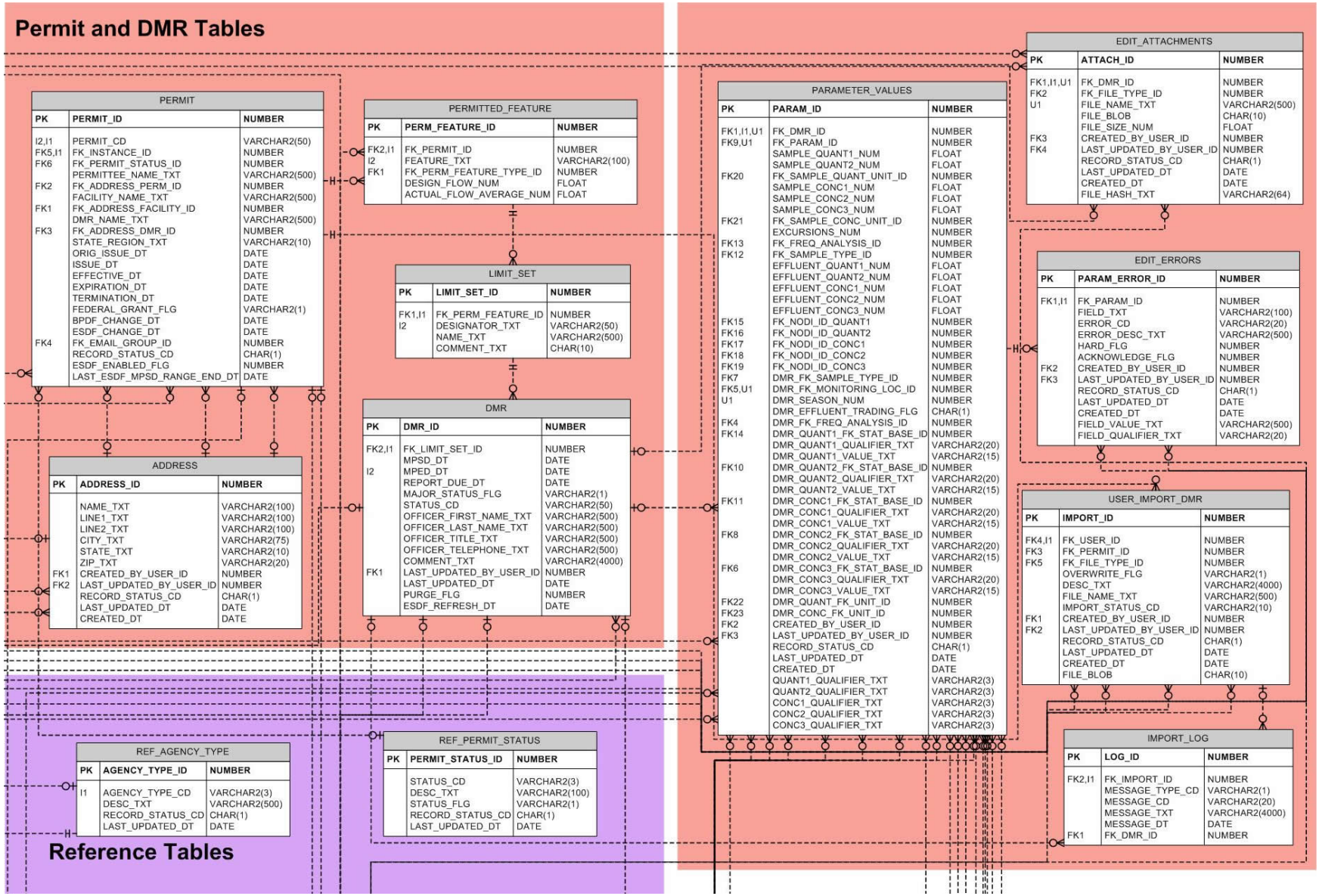

Figure 11-2. NetDMR Database Design Part 2

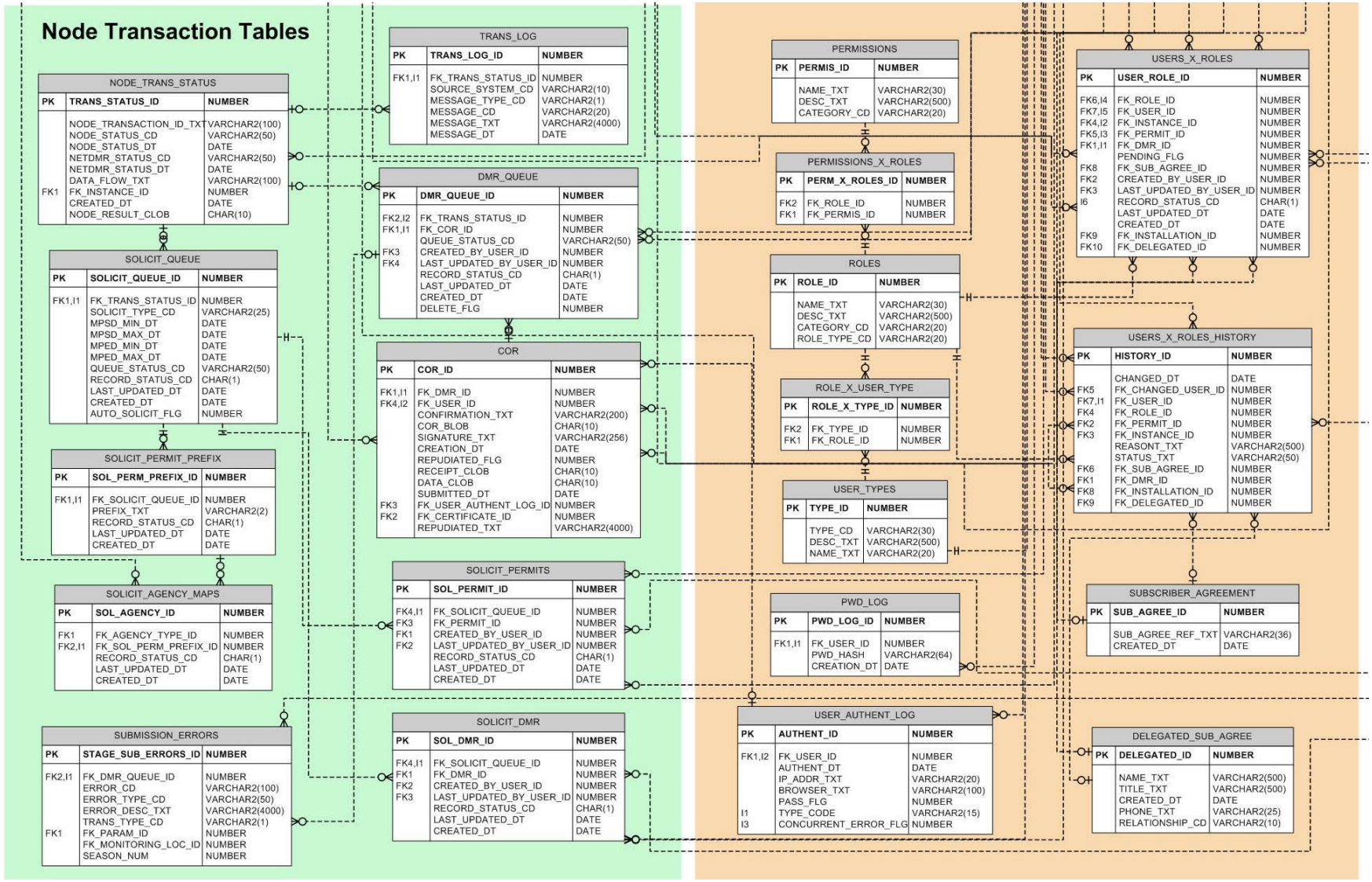

Figure 11-3. NetDMR Database Design Part 3

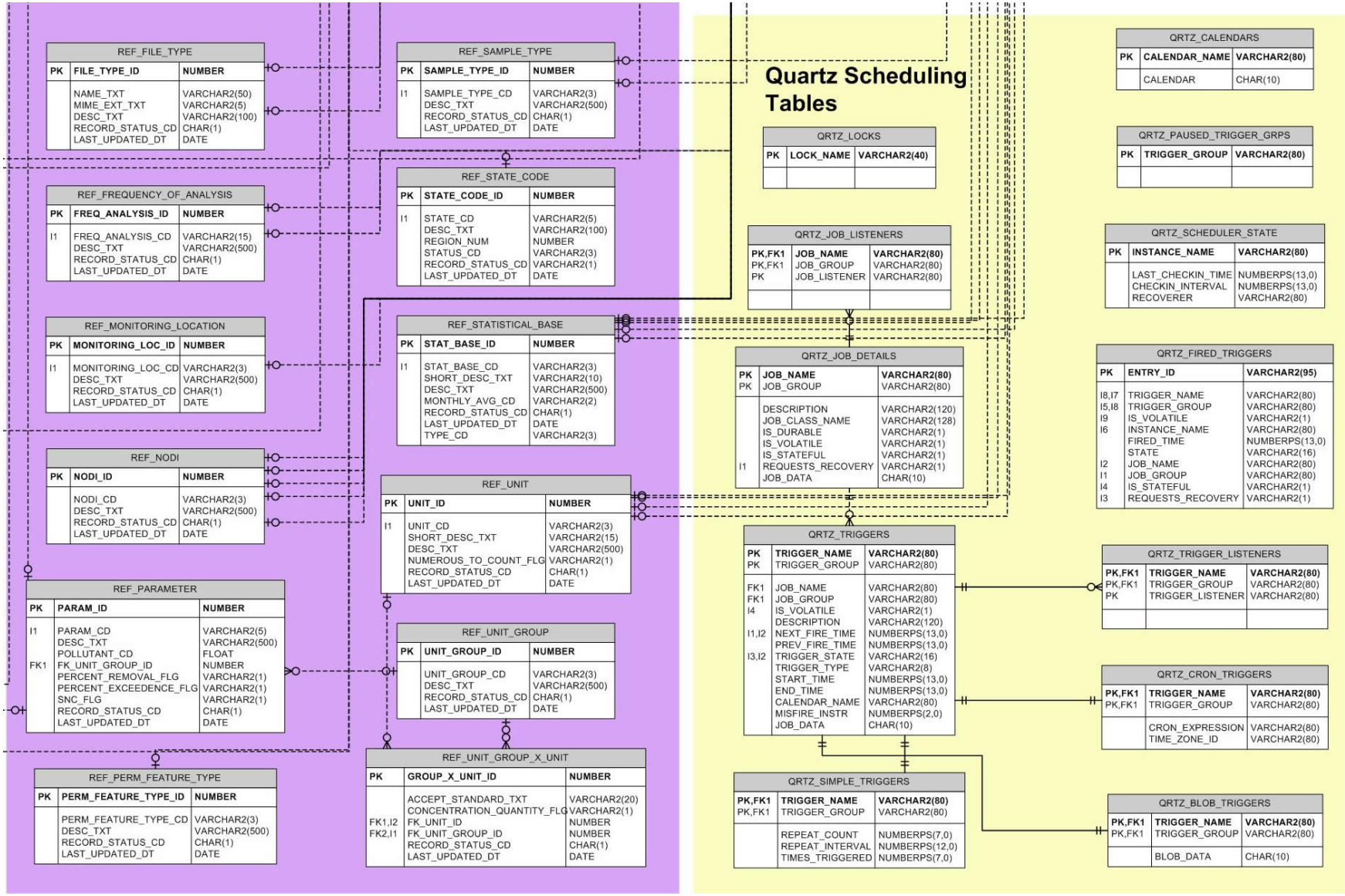

Figure 11-4. NetDMR Database Design Part 4