Establishing the Utah EPHT Node

- To date Utah EPHT has moved both air monitoring and drinking water data in large quantities from Utah Department of Environmental Quality (DEQ) to EPHT using proprietary software.
- Utah EPHT has also moved example data and JPG images from DEQ using a client version of their NEIEN node

Questions To Be Resolved During State FY 2007

- Will the client node version of the DEQ NEIEN node continue to be developed and supported? As the Utah EPHT node is developed out of code which implements the DEQ server node, will there by two nodes, one copy at DEQ and one at EPHT?
- Or will there just be one node (at DEQ), with proprietary exchange of data between EPHT and DEQ?

Considerations 1

- Data movement between EPHT and DEQ has been one way to date (DEQ to EPHT) using mostly proprietary Powerbuilder/Infomaker data pipeline objects.
- Such movement is highly efficient but is not considered optimally secure by DEQ, primarily because it may occur without DEQ's knowledge.

Considerations 2

- EPHT and DEQ have a well established trading partner agreement to exchange data – both ways.
- DEQ understands that good public health outcomes are central to DEQ's mission.
- Concerns remain at DEQ regarding when their data is quality assured (QA) enough to release to EPHT
- For example Air Quality data, before submission to EPA is not regarded as QA'ed.

Considerations 3

- Most likely as the DEQ node is rebuilt to add EPHT functionality, client functions will be included within it.
- With the final node built there will be one copy (at DEQ only) with Powerbuilder pipeline links to EPHT or two copies, with one each at DEQ and EPHT?
- Means must be established to have the resultant node serve EPHT's local health department clients.

Current Status of DEQ NEIEN Node Part 1

- The Utah NEIEN node is part of an XML web services network.
- Such networks are expected to become the preferred method for routine inter-governmental transfers of data.
- The DEQ node up to this time has been largely developed by a private contractor – Comsys of Salt Lake City.

Current Status of DEQ NEIEN Node Part 2

- The development language chosen was Powerbuilder.
- The database engine is Oracle.
- The web service application server is Powerbuilder's EAServer (Enterprise Application Server).
- Utah's EAServer realease is Windows based with a Linux front end to protect it.

Current Status of DEQ NEIEN Node Part 3

- The DEQ NEIEN node was initially certified with the EPA central data exchange (CDX) in December 2004.
- In February 2005 DEQ hired a full time node programmer/project manager.
- The node was recertified in December 2005, though not yet fully operational.

Data Flows Requested By EPA

- Air Quality
- Drinking Water
- Ambient Water Quality
- Environmental Response and Remediation (Toxic Release Inventory Exchange)
- Solid and Hazardous Waste (Resource Conservation and Recovery Act Information Exchange – RCRA)
- Facility Registry
- Homeland Security System Exchange
- Substance Registry System Exchange
- Biodiversity Data Exchange

Current NEIEN Node Architectural Overview

- In the NEIEN network nodes are intended to be “full” nodes, i.e. they are to be both clients and servers depending on the intent of the moment.
- The Utah DEQ node is in a “dynamic” state of completion.
- NEIEN network message exchanges are always XML encoded.

Current NEIEN Node Architectural Overview Continued

- Network messages must always conform to network defined XML schema.
- Errors generated are always SOAP errors.
- Lengthy attachments are always DIME encapsulated.
- The NEIEN network requires Secure Socket Level connections.
- Accordingly, a certificate has been secured and installed.

Current NEIEN Node Architectural Overview Continued

- As a measure of protection, the Utah DEQ node is front-ended with an Apache Web Server associated with a redirector which redirects received XML messages to the EAServer.
- This provides an extra layer of security as to internet visibility.
- The redirector is part of the EAServer distribution package.

Current NEIEN Node Architectural Overview Continued

- When a web service request from the world wide web is received by the Apache Web Server, the request is passed to the EAServer behind the utah.gov firewall.
- The EAServer looks up the service in the deploy.wsdd file.
- If the request is supported, the EAServer passes it to the UTDEQNEIEN Java wrapper class.

Current NEIEN Node Architectural Overview Continued

- The java wrapper recognizes the request as one of the Powerbuilder based web service components pushed on the EAServer using the Powerbuilder Eclipse tool.
- When the components are pushed, EAServer creates the web service and the “webserviceoperations” Java classes which are CORBA IDL links to the web service target in the Powerbuilder code.
- This web service Powerbuilder target includes a method for each primitive web service supported.

Primitive Web Services Supported By The Utah DEQ NEIEN Node

- The primary primitive web services supported are AUTHENTICATE, SOLICIT, and SUBMIT.
- Primitive Web Services: AUTHENTICATE, SOLICIT, SUBMIT, DOWNLOAD, GETSERVICES, GETSTATUS, NODEPING, NOTIFY, and QUERY.

Purposes of Primitive Web Services

- AUTHENTICATE – Authentication is part of the NEIEN security protocol.
- SOLICIT – The SOLICIT request calls for a named SQL statement or stored procedure to be executed on the server side, that is, the procedure resides on the server. It is intended for queries that may take a long time.

Purposes of Primitive Web Services Continued

- SUBMIT – The SUBMIT service provides for the transfer of a result set across the network, usually in response to a SOLICIT but it could also be an unSOLICITed SUBMIT.
- DOWNLOAD – The DOWNLOAD service requests transmission of a result set that is ready for exchange. The transaction “ready” status may have been passed by a GETSTATUS or a NOTIFY.
- GETSERVICES – The GETSERVICES request provides a node running as a client with a method of obtaining information about the services a server provides

Purposes of Primitive Web Services Continued

- QUERY – The QUERY request calls for a named SQL statement or stored procedure to be executed. The statement or procedure resides on the server. It is typically used for smaller information requests.
- NOTIFY – The NOTIFY service provides a method by which a node can broadcast a network message announcing result set availability.

Purposes of Primitive Web Services Continued

- GETSTATUS – The GETSTATUS service provides a node running as client with a method of determining the current status of a previous SOLICIT, SUBMIT, or NOTIFY.
- NODEPING – The NODEPING service is the network method of determining if a node is responsive.

Utah State FY 2007 Considerations

- Current Utah DEQ NEIEN node Powerbuilder code will provide the core of the Utah EPHT node.
- If there is only one combined DEQ/EPHT node, data will be moved to and from that node using Powerbuilder pipeline objects.
- If Utah EPHT has its own node it will likely use unused capacity on the EAServer.
- EPHT metadata storage, handling, and processing must be added.
- Means of supporting Utah EPHT's health department clients must be developed.

Simulation of National EPHT Network

- In developing Utah EPHT node functionality it would be useful to develop a prototype node to serve as a stand in for the national central EPHT node, for test purposes.

Creating a Web Interface For Local Health Department Clients

- The EAServer application server includes an Apache web server.
- Java Server Pages (JSP's) can be readily created from within enterprise Powerbuilder, as an implementation of the J2EE application model.
- JSP web page creation proceeds in a manner similar to Dreamweaver.

Advantages For Utah EPHT In Using The Utah DEQ Node As A Base For Its Own Node – A Consideration For Others

- The Utah DEQ node includes a large mass of reusable code, which Utah EPHT doesn't have to write.
- Utah DEQ has programming expertise useful to EPHT
- The Powerbuilder environment supports ready access to a wide range of proprietary databases.
- Powerbuilder code is Basic based.
- CDC and DEQ-partner EPA are already collaborating.

Utah DEQ Deployment View

There are only 9 messages that NEIEN nodes respond to. They are

Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services

AUTHENTICATE Web Service

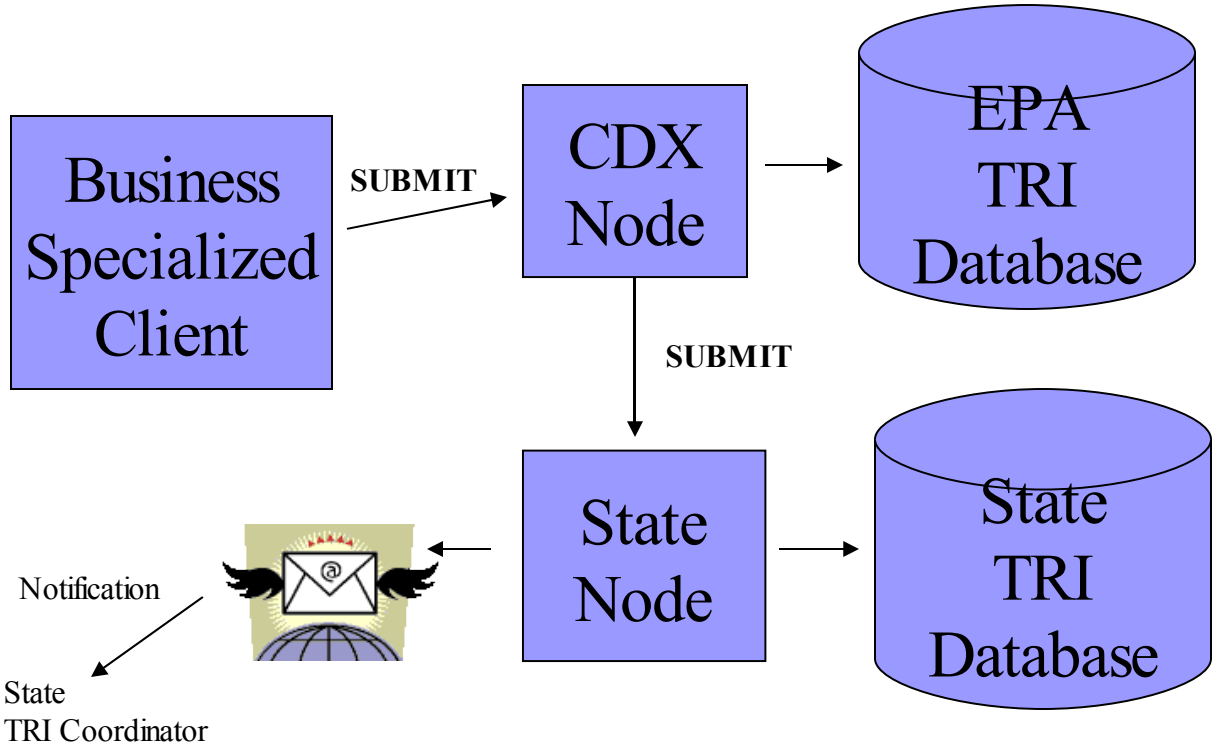
Before a client can do anything else it must **AUTHENTICATE** to receive permission to use the network by passing credentials and asking the **N**etwork **A**uthentication and **A**uthorizing **S**ervice (NAAS) node for a security token.

Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services

SUBMIT Web Service

In the Toxic Release Inventory (TRI) exchange, a business uses a specialized client to **SUBMIT** a report to the EPA. As soon as the Central Data Exchange (CDX) node receives such a report it immediately reflects it to the state node using the **SUBMIT** web service. This saves the business double reporting and saves the state reconciliation with the EPA.

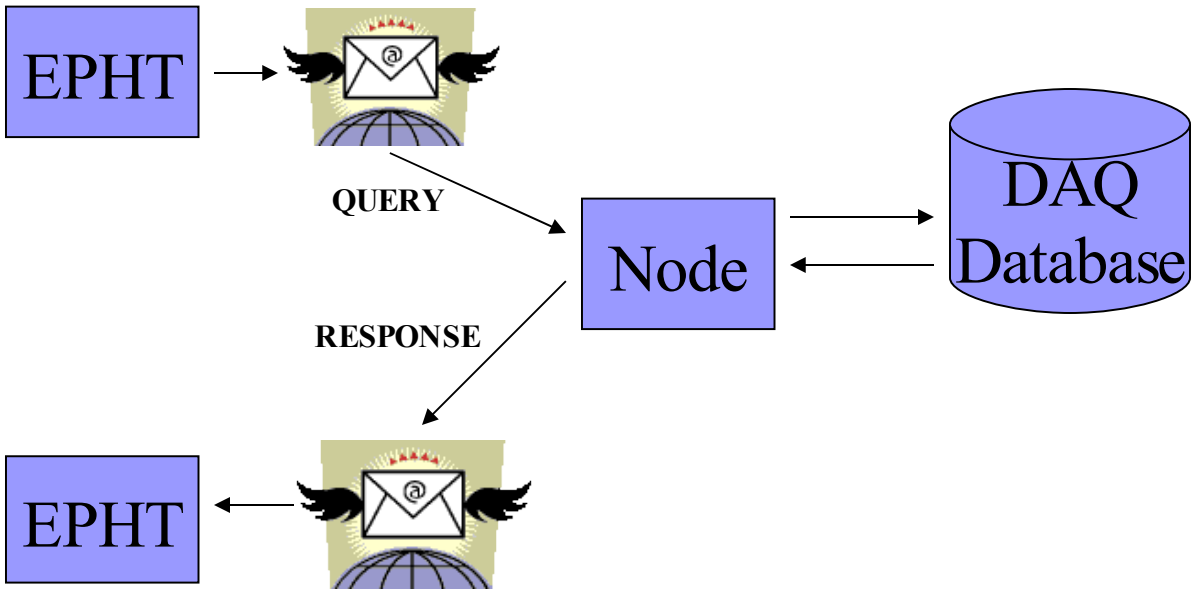
Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services



QUERY Web Service

The DOH Environmental Public Health Tracking project (EPHT) would like to obtain air data reports from DAQ using the node. One approach would be for EPHT to email their **QUERY** request to the node. The node would then perform the query and return the results as an Excel spreadsheet attachment in a response email.

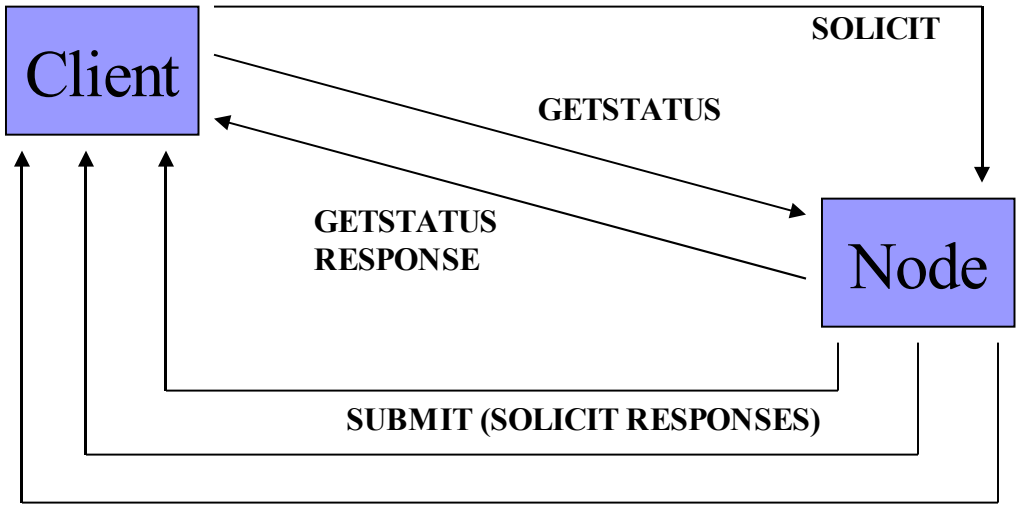
- Authenticate
- Submit
- **Query**
- Get Status
- Notify
- Solicit
- Download
- Node Ping
- Get Services



SOLICIT/GETSTATUS Web Services

It may become desirable to obtain the entire EPA FRS facility database to compare with our own . The **SOLICIT** web service was created to handle large data sets that may involve multiple packet responses. If a great deal of time is consumed and the client user wants a progress report, they may use **GETSTATUS** to determine the progress of the **SOLICIT** response.

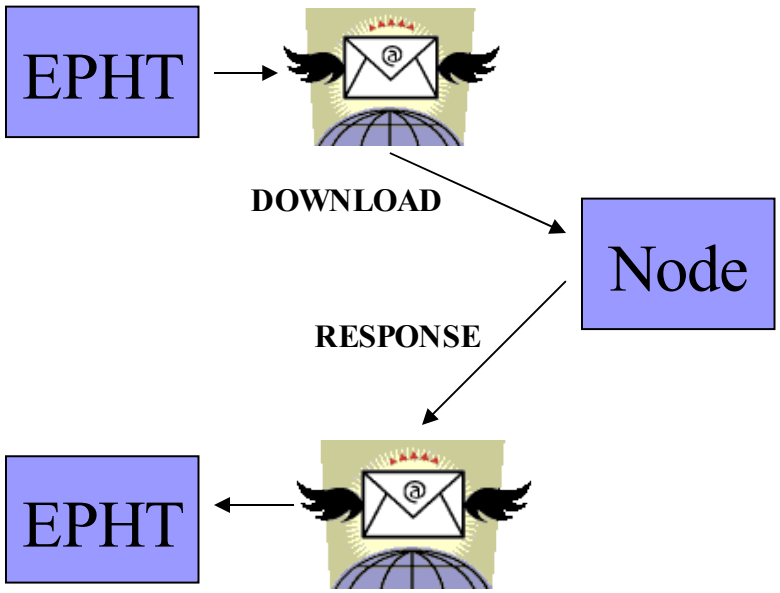
Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services



DOWNLOAD Web Service

When the CDX reflects a TRI report to the state, the node automates the loading of that report into the state TRI database. However, someone from DERR may wish to view that report as the node received it in standard report form. They might use their email to issue a **DOWNLOAD** request to the node and receive the TRI report as a response email attachment.

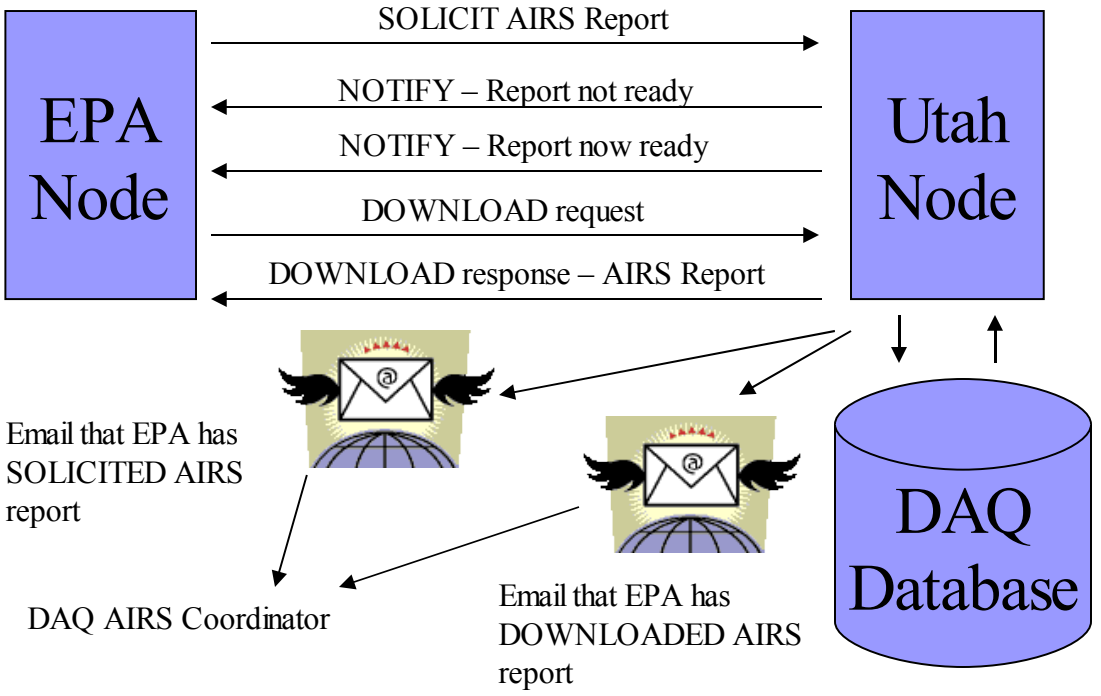
- Authenticate
- Submit
- Query
- Get Status
- Notify
- Solicit
- **Download**
- Node Ping
- Get Services



NOTIFY Web Service

The EPA node may have an automated **SOLICIT** for the AIRS report from DAQ before it is ready. The Utah node would respond with a **NOTIFY** that the report was not ready. Then, at some later time, the Utah node would **NOTIFY** the EPA node that the report was ready and the EPA node would **DOWNLOAD** the report.

Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services



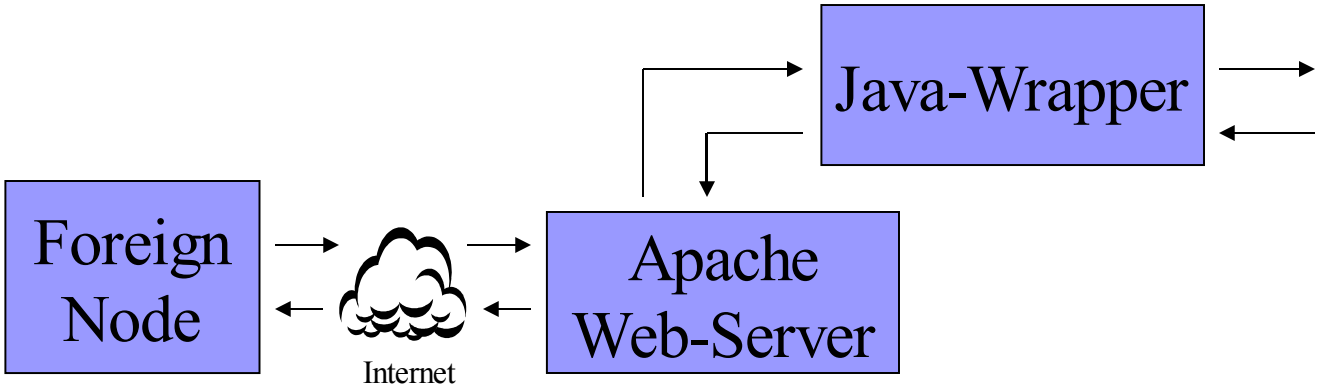
XML

The NEIEN nodes also use XML “schemas” to validate exchanged data. If the XML data does not match the XML schema then the data is rejected. The XML schema is a QA/QC tool. This is a portion of the NEIEN CDX XML schema.

```
<message name="Authenticate">
        <part name="userId" type="xsd:string"/>
        <part name="credential" type="xsd:string"/>
        <part name="authenticationMethod" type="xsd:string"/>
</message>
<message name="AuthenticateResponse">
        <part name="return" type="xsd:string"/>
</message>
<message name="Query">
        <part name="securityToken" type="xsd:string"/>
        <part name="request" type="xsd:string"/>
        <part name="rowId" type="xsd:integer"/>
        <part name="maxRows" type="xsd:integer"/>
        <part name="parameters" type="typens:ArrayOfstring"/>
</message>
<message name="QueryResponse">
```

NODE DESIGN

The several states have used a variety of tools to build their nodes depending on their shop standards. The Microsoft ".NET" package is very popular. The Utah node web services are written in Sybase Powerbuilder, a rapid application development (RAD) language. These web service components are deployed to a Sybase EAServer Application Server. Utah built a thin Java wrapper to translate network protocol terms that are protected keywords in Powerbuilder.



Sybase EAServer Application Server

9 Services

Authenticate
Submit
Query
Get Status
Notify
Solicit
Download
Node Ping
Get Services