Oregon ALERT IIS

HL7 Real Time Data Exchange

Version 1.3

Last Updated: June 11, 2012

Note: HL7 2.4 in this document refers to the local (Oregon) implementation of the CDC HL7 2.3.1 (June 2006) Guide

Introduction

Thank you for your interest in Health Level Seven (HL7) electronic data exchange with Oregon ALERT IIS.

HL7 real time data exchange with ALERT IIS is accomplished by sending via SOAP web services. For information about sending HL7 batch, please contact the ALERT Help Desk.

The first step for a clinic and/or their vendor is to review the ALERT IIS HL7 Implementation Guide. It is important to understand the local HL7 specifications to ensure that your EHR system can prepare and send HL7 messages that meet Oregon ALERT's requirements.

Once you have reviewed the ALERT IIS HL7 Implementation Guide and the following data exchange specifications, contact the ALERT Help Desk to begin the process of setting up a data exchange relationship with ALERT IIS. Once a site has communicated to the Help Desk that they are interested in setting up HL7 real time data exchange with ALERT, a data exchange specialist will follow up.

The ALERT IIS HL7 Implementation Guide can be downloaded from our website:
https://www.alertiis.org/docs/hl7_24_gts.pdf

For questions or more information:

ALERT Help Desk
1-800-980-9431
alertiis@state.or.us

Checklist for Provider Organizations and Vendors Establishing Real Time Data Exchange

1) Planning & Design

   - ☐ Review ALERT IIS HL7 2.4 Implementation Guide. Please note there are a number of Oregon-specific HL7 requirements. Two of those are:
     - Clinic level code (AL#, provided by ALERT) should be sent in MSH-4, and RXA-11 (if you plan to use the inventory module).
     - Vaccine eligibility code (for VFC program or other state supplied vaccine) must be sent at the dose level (in an OBX segment), rather than at the patient visit level.
   - ☐ Review current list of sites in ALERT IIS associated with your organization. Add/modify as needed. *This list will be provided by the ALERT Data Exchange Coordinator.*
   - ☐ Test plans: begin writing test plans and set up test cases/patients in your test environment.

2) Development

   - ☐ Per the ALERT IIS HL7 2.4 Implementation Guide, map your EHR system codes to the specifications described for ALERT IIS. There are various tools available online for validating standard HL7 message format, such as https://phinmqf.cdc.gov/.
   - ☐ Develop VXU message format, per the ALERT IIS specifications.
   - ☐ Develop VXQ message format, per the ALERT IIS specifications. In planning for bi-directional query functionality, please consider the following questions in your planning and development:
     - Will records be stored permanently?
     - If storing, will returned records be integrated into the EHR immunization history?
     - Will the EHR automatically query for records, or will the user initiate the query?
     - How will the returned records be displayed to the user?
   - ☐ Error Message and Response File Management. Please consider the following questions in your planning and development:
     - How will the EHR manage error messages?
     - How will error messages be displayed to the user?
     - Who will review error messages?
     - How will providers be able to correct errors and re-submit?
     - If there is web service downtime for maintenance or outage, what is the plan for how records will be re-sent?

3) Check In Call with ALERT IIS Data Exchange Team

    - The purpose of this call is to review design and development steps, determine readiness and review next steps for testing and deployment. Please contact Tracy Little at tracy.c.little@state.or.us

4) Testing

    - ☐ Install SSL Certificate for access to ALERT UAT (ALERT IIS test environment), from your test environment. See detailed instructions in *SSL Certificate Creation v1.3.*
    - ☐ Install SOAP web services WSDL (web services definition language) for UAT environment. The WSDL file will be provided by the ALERT IIS Data Exchange Coordinator.
    - ☐ Submit one day, then one month of immunization data to the ALERT IIS UAT environment via the web service.
    - ☐ Conduct internal QA and validation of test messages. Suggested review criteria are as follows:
        - Find out how many successful, partially successful, and failed VXU messages are sent.
        - Identify major issues in failed and partially successfully messages; submit another set (month) of data until issues are resolved. Repeat this step until nearly all messages are successful.
            - ▪ Check the User Interface to ensure that the immunizations reported in the messages were added to the patients' records.
            - ▪ Check the messages and UI to ensure that historical immunizations and recommended fields are being sent.
            - ▪ Run the percentage of each vaccine and group reported during the one-month test. Ensure that percentages do not vary greatly from the amount of reporting we normally expect.
    - ☐ Review results of testing with ALERT IIS Data Exchange Coordinator. Schedule Go-Live.

5) Deployment

    - ☐ Install SSL Certificate for access to ALERT IIS production site, from your production environment. *The same certificate will be used for UAT and production; follow the same steps completed earlier while connecting to the UAT environment.*
    - ☐ Install the SOAP web services WSDL (web services definition language) for Production environment.
    - ☐ Coordinate date for go-live with ALERT data exchange coordinator. If currently sending electronic data to ALERT plans must be made to suspend current submission by go-live date.
    - ☐ Validate data going to ALERT IIS production site and review with ALERT IIS Data Exchange Coordinator.

SSL Certificate Creation

The Secure Sockets Layer (SSL) is a commonly-used protocol for managing the security of a message transmission on the Internet. SSL client and server certificates are used as an added security feature for transmitting data for Alert IIS web services transactions. We require that both the client and server install certificates that have been generated by the HP Immunization Services personnel. To accomplish this, you must first create a private key for each machine that will be accessing the Alert IIS web services machine. This private key is then used to create a Certificate Signing Request (CSR) which will be sent to HP. HP will create the SSL certificate which will be returned to you for installation on your client machine ("client" in this instance will most likely be the server that communicates with the Alert IIS web services servers).

Generating a Key and Certificate Signing Request (CSR)

To generate a CSR, a key pair must be created for the server. These two items are a digital certificate key pair and cannot be separated. If the public/private key file is lost or changed before the SSL certificate is installed, the SSL certificate will need to be re-issued. The private key, CSR, and certificate must all match in order for the installation to be successful. The following sequence of commands will generate a 2048 bit key using the OpenSSL software. Below that are instructions for creating the CSR in a Windows environment. We recommended the use of the domain name or IP address that will be used for the certificate as the prefix of the filenames. Also make sure that any existing keys and CSR's are NOT overwritten.

Generating a Private Key and CSR using OpenSSL

Step 1: Generate Private Key

Type the following command at the prompt:

```
        openssl genrsa –out my.server.com.key 2048
```

This command generates a 2048 bit RSA private key and stores it in the file, my.server.com.key

Note: For all SSL certificates, the CSR key bit length must be 2048.

Step 2: Generate the CSR

Type the following command at the prompt:

```
openssl req –new –key my.server.com.key –out my.server.com.csr
```

This command will prompt for the following attributes of the certificate:

Field	Required / Optional	Description
Country Name	R	Use the two-letter code without punctuation for country. Example: US or CA
State or Province	R	Spell out the state completely; do not abbreviate the state or province name. Example: Oregon
Locality or City	R	The Locality field is the city or town name; do not abbreviate. (Example: Saint Louis, not St. Louis)

Company	R	If the company or department has an &, @, or any other symbol using the shift key in its name, the symbol must be spelled out or omitted. Example: XY & Z Corporation would be XYZ Corporation or XY and Z Corporation.
Organizational Unit	O	Can be used to help identify certificates registered to an organization. The Organizational Unit (OU) field is the name of the department or organization unit making the request. To skip the OU field, press Enter on the keyboard.
Common Name	R	The Common Name is the Host + Domain Name. It looks like "my.server.com".

Certificates can only be used on Web servers using the Common Name specified during enrollment. For example, a certificate for the domain "server.com" will receive a warning if accessing a site named "www.server.com" or "secure.server.com", because "www.server.com" and "secure.server.com" are different from "server.com".
Note: Please do not enter an email address, challenge password or optional company name when generating the CSR.
A public/private key pair has now been created. The private key (my.server.com.key) is stored locally on the server machine and is used for decryption (DON'T LOSE IT). The public portion, in the form of a Certificate Signing Request (my.server.com.csr), will be for certificate enrollment.

Step 3: Send CSR to ALERT

The CSR is an ASCII text file that can be attached to an email and should be sent to the ALERT IIS Data Exchange Coordinator. Please send your CSR to:
Tracy Little (tracy.c.little@state.or.us)

Step 4: Backup the private key

It is recommended that you back-up the *.key* file. A good choice is to create a copy of this file onto a diskette or other removable media. While backing up the private key is not required, having one will be helpful in the instance of server failure.

Step 5: Receiving Your Signed Certificate

Once HP is done processing your CSR, you will receive the signed certificate as an email attachment. This file must be installed in the Trusted Store of the computer for which it was generated. The instructions for this will vary depending on your environment. A good source of reference for this information is Google (search for: importing trusted root certificates). Two common trusted stores are Public-Key Cryptography Standards #12 (PKCS#12 or PFX) and Java Key Store (JKS).

Join your private key with the signed certificate and certificate authority files mailed by HP, so the browser and OS can use it.

Here is an example of creating a .pfx file using openssl.

```
Openssl pkcs12 –export –out www.example.com.pfx –inkey www.example.com.key –
in www.example.com.crt –certfile cacert.crt
```

Here is what the example file names represent:

www.example.com.pfx = this will be the output file – which you'll install into Windows 7 so IE can use it, etc

www.example.com.key = this is the key that was generated by step 1
www.example.com.crt = this is the signed certificate provided in response to your CSR
cacert.crt = this is the CA (Certificate Authority) file which was provided, this is needed by openssl to verify we truly signed the first file.

Generating a Private Key and CSR using Microsoft Windows

Step 1: Creating the Private Key and CSR

1. Open the **Microsoft Management Console** (MMC). On the Start menu, click Run, type MMC, and then click OK. MMC opens with an empty console.
2. Right-click the default Web site, click **New**, and then click **Site**. Create a new site and give it a temporary name.
3. Right-click the new site, click **Properties**, click the **Directory Security** tab, and then click **Server certificate**.
4. Select **Create new certificate** and follow the wizard to create a new CSR. Use the information from the OpenSSL instructions above in Step 2, when filling out the request. When prompted, select **Prepare the request now but send it later**.
5. Use the CSR that you just created to request a new certificate from HP.

See the OpenSSL instructions Steps 3, 4, and 5, above, to finish the process.

SSL Trust Stores in a Web Services Context

Keystores

A keystore is a database of private keys and their associated X.509 certificate chains authenticating the corresponding public keys. A key is a piece of information that controls the operation of a cryptographic algorithm. For example, in encryption, a key specifies the particular transformation of plain text into ciphertext, or vice versa during decryption. Keys are used in digital signatures for authentication.

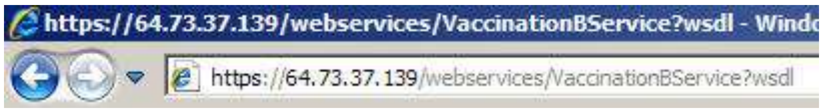
Truststores

The truststore contains the Certificate Authority (CA) certificates and the certificate(s) of the other party to which this entity intends to send encrypted (confidential) data. This file must contain the public key certificates of the CA as well as the client's public key certificate.

Alert IIS Web Services Client Operations

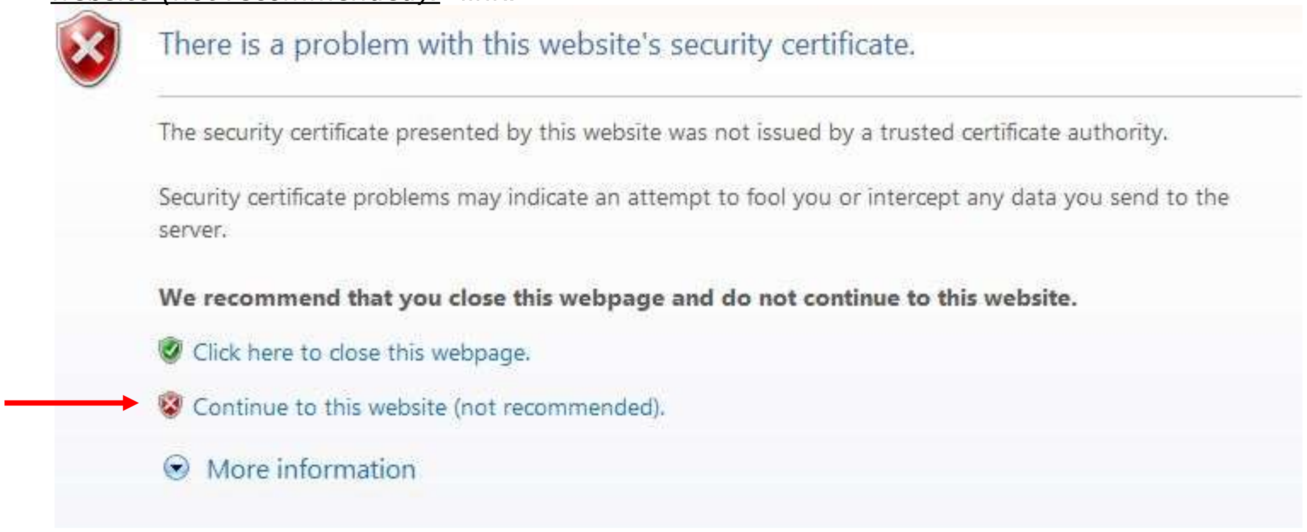
For all provider servers which will be accessing the ALERT IIS Web Service, a Truststore needs to be established. The Truststore should contain the signed certificate you received from HP plus the CA certificate(s) from the Web Services server.

Disclaimer: *The following screen shots were taken using* ***Internet Explorer 7*** *and may differ depending upon the browser you are utilizing.*
The CA certificate(s) can be acquired, by pointing your web browser at the ALERT IIS Web Services site (e.g., using a WSDL request).


https://64.73.37.139/webservices/VaccinationBService?wsdl - Windo

https://64.73.37.139/webservices/VaccinationBService?wsdl

Notice: *You may receive the following notification "There is a problem with this website's security certificate." If you do receive this message, please, click the "Continue to this website (not recommended)." link.*


There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.
Continue to this website (not recommended).
More information

Right click on the browser's lock icon to display the certificate. If the lock icon is not present you may see, click on Certificate Error.

Get information about Secure Sockets Layer (SSL) certificates

When you connect to a website, Internet Explorer uses a secure connection that uses Secure Sockets Layer (SSL) technology to encrypt the transaction. The encryption is based on a certificate that provides Internet Explorer with the information it needs to communicate securely with the website. Certificates also identify the website and owner or.
You can view a certificate to validate the identity of a website before providing information.

To validate the identity of a website:

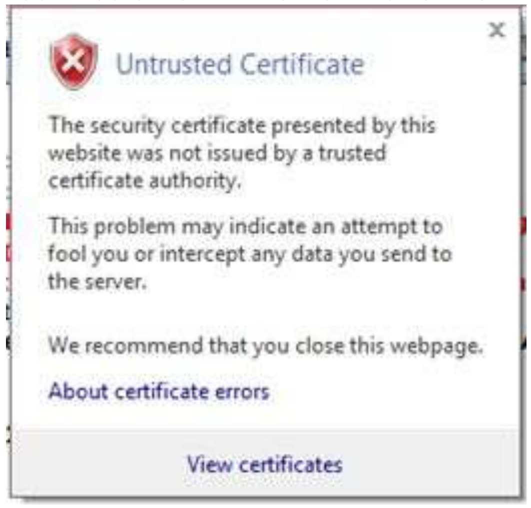
1. Open Internet Explorer
2. Go to the website that you want to validate.
3. Click the Lock icon 🔒, which is located to the right of the Address bar.

   Basic certificate information (for example, the name and address of the website owner and information about who certified the site) will be displayed. To see additional information, click View Certificates.

Note:

If a lock icon does not appear in the Address bar in step 3 above, the connection is not secure.

You should now see a notification "Untrusted Certificate". Click on "View certificates"



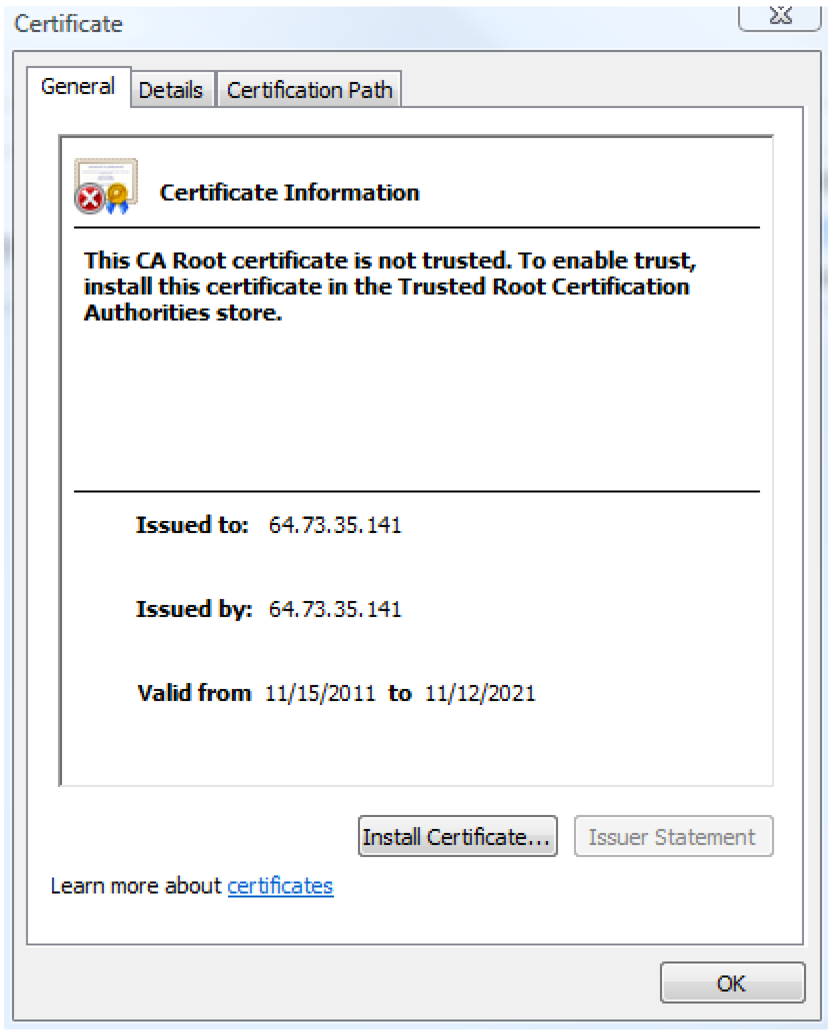
Untrusted Certificate

The security certificate presented by this website was not issued by a trusted certificate authority.

This problem may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage.

You should now be presented with the Certificate dialogue box.



Certificate Information

This CA Root certificate is not trusted. To enable trust, install this certificate in the Trusted Root Certification Authorities store.

Issued to: 64.73.35.141

Issued by: 64.73.35.141

Valid from 11/15/2011 **to** 11/12/2021

Depending on the browser model and version which you are using, you will now be able to import the CA certificates into your machine's truststore, or export them for importing into a file by openssl.

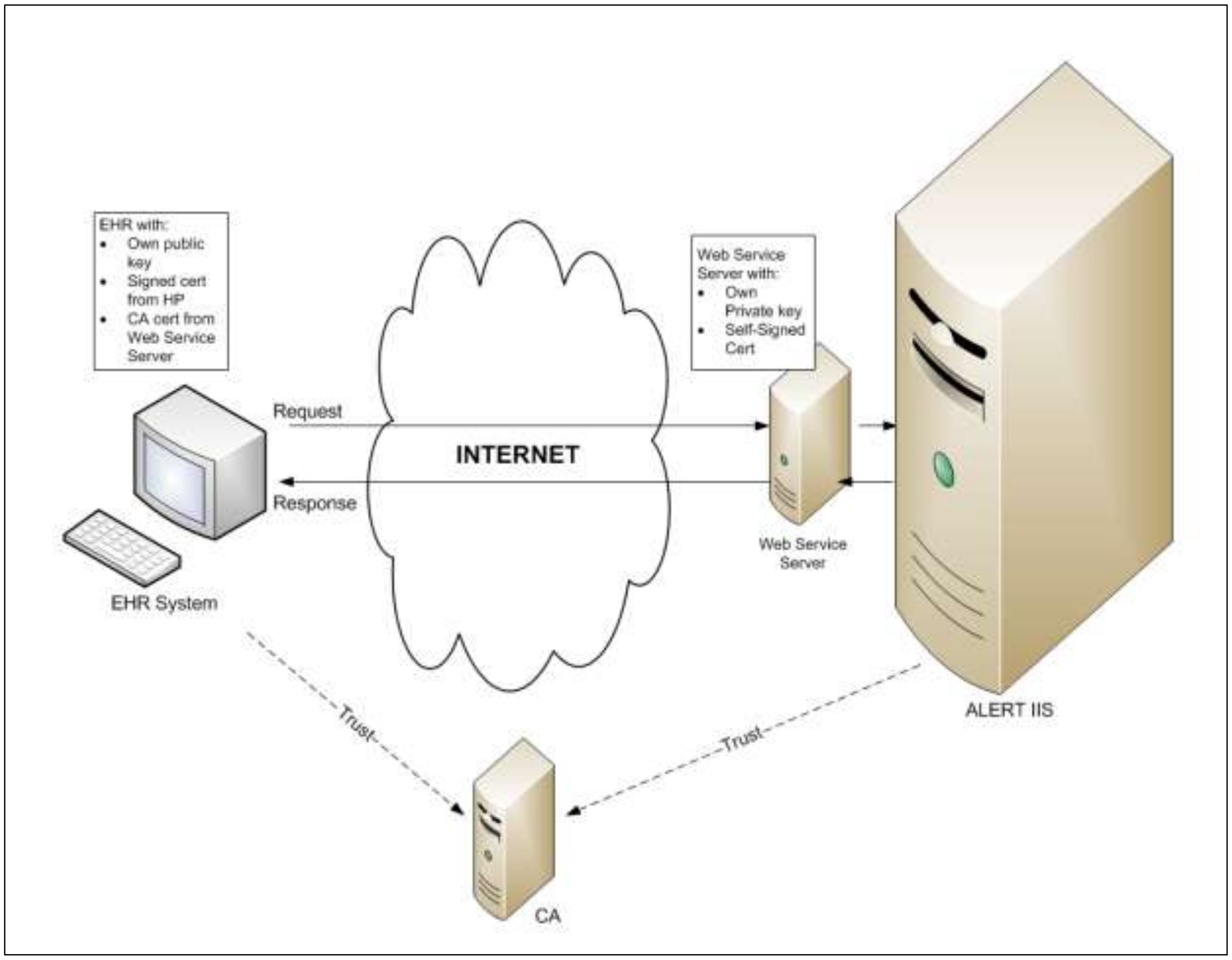
Whether you need to import the machine's truststore or must create a truststore file is determined by the type of Web Service Client you are creating and should be explained in your documentation for that system.

Please note: The IP address noted in the picture above for Issued to and Issued by is in name only. Be careful not mistake the Issuers name for the IP address of the current WSDL.

Introduction to SOAP Web Services in ALERT IIS

A Web Service is a standards-based method of allowing one computer to access functions in another computer through the Internet. The functions are accessed through the same ports used by Internet browsers, making them very likely to be allowed through firewalls.

Security is implemented by installing public and private key pairs (known as X.509 certificates) on both the ALERT IIS Web Services Server and the Electronic Health Record (EHR) computer requesting the information. Using this technique, both the client and server are identified to each other to establish trust and the communication is encrypted.



ALERT IIS supports the following functions through Web Services:

1. FindHistory, used to query the IIS for immunization data
2. UpdateHistory, used to send information about an administered vaccine

Web Service functions are called using Simple Object Access Protocol (SOAP) requests, which are formatted as XML (eXtensible Markup Language) messages.

Each SOAP request is made up of the following elements:

1. The envelope, which identifies the message as a SOAP request

2. The function name
3. The parameters of the function call. In most cases, the parameters are individual pieces of data, such as First Name, Last Name, etc. For the IIS Web Services, there is a single parameter which contains an entire HL7-formatted message, wrapped in a CDATA section to keep it from being misinterpreted by the parser.

Here is a simple VXU message sent to the UpdateHistory Web Service function. The pieces are numbered from the list above.

(1) SOAP Envelope Start	<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:vac="http://vaccination.org/"> <soap:Header/> <soap:Body>
(2) Function and parameter Start	<vac:UpdateHistory> <arg0>
CDATA section Start	<![CDATA[
(3) Parameter, the HL7 message, segments vary depending on the function	MSH|^~\&||AL9999||ALERTIIS^^^|20110201||VXU^V04|682299|P^|2.4^^|||ER PID|||79928^^^^PI|A5SMIT0071^^^^^|SMITH^MAY^^^^^^^|JOHN^^^^^^^^|20101212|F|||| RXA|0|999|20110201|20110101|^^^90701^DTP^CPT|0.5
CDATA section End	]]>
(2) Function and parameter End	</arg0> </vac:UpdateHistory>
(1) SOAP Envelope End	</soap:Body> </soap:Envelope>

The definitions of the functions are specified in the WSDL (Web Services Definition Language) file. Modern development environments (such as Java and .NET) can take the WSDL and turn it into a programming interface to simplify implementation. The WSDL can be supplied by the Oregon Immunization Department or retrieved from the server once the certificates are installed.

For More Information
http://www.w3schools.com/webservices/default.asp is a good source for information about Web Services. The Summary page includes links to information about WSDL and SOAP.

Getting the WSDL

The WSDL (Web Services Definition Language, pronounced *whiz-dul*) is a method used to describe Web Services functions that can be accessed through the Internet from other computers.

The contents of the WSDL and related XSD file are included in Appendix A, for reference.

Using the WSDL file, modern development environments (such as Java and .NET) can turn the WSDL file into a programming interface to simplify implementation.

The WSDL can be obtained in three ways:

1. Retrieve it from the Web Service Server
2. Get a copy from the Oregon Immunization Department
3. Extract the UAT version from this document

Method 1: Retrieve the WSDL from the Web Service Server

Prerequisites: This method requires that you have already requested, received and installed the certificates to reach the Web Server with full authentication and encryption.

This address can be used to retrieve the WSDL into a web browser or a development tool to generate a programming interface. The steps for working with the programming interface are beyond the scope of this documentation.

1. Enter the appropriate address into the browser address bar or the development tool.

UAT:	https://64.73.37.139/webservices/VaccinationBService?wsdl
Production:	https://soa.alertiis.org/webservices/VaccinationBService?wsdl

   If you leave off the trailing "?wsdl" you will get a page of endpoints, rather than the WSDL itself.

2. The WSDL appears in the web browser and can be printed for reference.

3. To get the related XSD file, do the same thing with this address. This will only be needed to view the files from a web browser, because most applications using a WSDL will automatically download and use the XSD file.

UAT:	https:// 64.73.37.139/webservices/VaccinationBService?xsd=1
Production:	https://soa.alertiis.org/webservices/VaccinationBService?xsd=1

4. The XSD appears in the web browser and can be printed for later reference.

Method 2: Get a copy from the Oregon Immunization Department

Prerequisites: None.

Contact the ALERT Help Desk (alertiis@state.or.us) and request electronic copies of the WSDL file. This will provide the initial IIS-WSDL-Dev.xml and IIS-WSDL-Prod.xml files, which contain the Web Service Definitions. These could also be copied out of this document, below, and pasted into text files.

Method 3: Extract the UAT version from this document

Prerequisites: None.

1. Highlight the lines in Appendix A (the WSDL section only) from
   `<?xml version="1.0" encoding="UTF-8" ?>`
   to
   `</definitions>`
2. Copy the highlighted text to the clipboard with Control-C.
3. Open up Notepad and paste the contents of the clipboard into the empty Notepad file.
4. Verify the contents include all of the lines you highlighted in Step 1.

Appendix A: WSDL and XSD File Contents

The following are the contents of the UAT version of the WSDL. The Production WSDL is identical, except for the location tags which show the Web Service endpoint. Specifically, “https://64.73.37.139” becomes “https://soa.alertiis.org” but the rest is identical.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.1-hudson-28-.   -->
<!--  Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.1-hudson-28-.   -->
<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:wsp="http://www.w3.org/ns/ws-
policy" xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:tns="http://vaccination.org/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://vaccination.org/" name="VaccinationBService">
    <wsp:Policy xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702"
wsu:Id="WS2007FederationHttpBinding_IVaccinationServiceBindingPolicy">
        <sp:SignedEncryptedSupportingTokens>
            <wsp:Policy>
                <sp:UsernameToken sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient">
                <wsp:Policy>
                    <sp:WssUsernameToken10 />
                </wsp:Policy>
            </sp:UsernameToken>
            </wsp:Policy>
        </sp:SignedEncryptedSupportingTokens>
        <sp:TransportBinding>
            <wsp:Policy>
                <sp:AlgorithmSuite>
                    <wsp:Policy>
                        <sp:Basic128 />
                    </wsp:Policy>
                </sp:AlgorithmSuite>
                <sp:IncludeTimestamp />
                <sp:Layout>
                    <wsp:Policy>
                        <sp:Lax />
                    </wsp:Policy>
                </sp:Layout>
                <sp:TransportToken>
                    <wsp:Policy>
                        <sp:HttpsToken RequireClientCertificate="false" />
                    </wsp:Policy>
                </sp:TransportToken>
            </wsp:Policy>
        </sp:TransportBinding>
        <sp:Wss10 />
        <wsam:Addressing />
    </wsp:Policy>
    <types>
        <xsd:schema>
            <xsd:import namespace="http://vaccination.org/" schemaLocation="https://64.73.37.139:443/webservices/VaccinationBService?xsd=1" />
        </xsd:schema>
    </types>
    <message name="UpdateHistory">
        <part name="parameters" element="tns:UpdateHistory" />
    </message>
    <message name="UpdateHistoryResponse">
        <part name="parameters" element="tns:UpdateHistoryResponse" />
    </message>
    <message name="Exception">
        <part name="fault" element="tns:Exception" />
    </message>
    <message name="FindHistory">
        <part name="parameters" element="tns:FindHistory" />
    </message>
    <message name="FindHistoryResponse">
        <part name="parameters" element="tns:FindHistoryResponse" />
    </message>
    <portType name="IVaccinationService">
        <operation name="UpdateHistory">
            <input wsam:Action="http://vaccination.org/IVaccinationService/UpdateHistoryRequest" message="tns:UpdateHistory" />
            <output wsam:Action="http://vaccination.org/IVaccinationService/UpdateHistoryResponse" message="tns:UpdateHistoryResponse" />
            <fault message="tns:Exception" name="Exception" wsam:Action="http://vaccination.org/IVaccinationService/UpdateHistory/Fault/Exception"
/>
        </operation>
        <operation name="FindHistory">
            <input wsam:Action="http://vaccination.org/IVaccinationService/FindHistoryRequest" message="tns:FindHistory" />
            <output wsam:Action="http://vaccination.org/IVaccinationService/FindHistoryResponse" message="tns:FindHistoryResponse" />
            <fault message="tns:Exception" name="Exception" wsam:Action="http://vaccination.org/IVaccinationService/FindHistory/Fault/Exception" />
        </operation>
    </portType>
    <binding name="WS2007FederationHttpBinding_IVaccinationServiceBinding" type="tns:IVaccinationService">
        <wsp:PolicyReference URI="#WS2007FederationHttpBinding_IVaccinationServiceBindingPolicy" />
        <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" style="document" />
        <operation name="UpdateHistory">
            <soap12:operation soapAction="" />
            <input>
                <soap12:body use="literal" />
            </input>
            <output>
                <soap12:body use="literal" />
                </output>
                <fault name="Exception">
                    <soap12:fault name="Exception" use="literal" />
                </fault>
            </operation>
            <operation name="FindHistory">
                <soap12:operation soapAction="" />
                <input>
                    <soap12:body use="literal" />
                </input>
                <output>
                    <soap12:body use="literal" />
                </output>
                <fault name="Exception">
                    <soap12:fault name="Exception" use="literal" />
                </fault>
            </operation>
        </binding>
        <service name="VaccinationBService">
            <port name="WS2007FederationHttpBinding_IVaccinationService" binding="tns:WS2007FederationHttpBinding_IVaccinationServiceBinding">
                <soap12:address location="https://64.73.37.139:443/webservices/VaccinationBService" />
            </port>
        </service>
    </definitions>
```

The following are the contents of the UAT XSD file, which is retrieved automatically when the main WSDL is imported into a development tool.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!--  Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.1-hudson-28-.   -->
<xs:schema xmlns:tns="http://vaccination.org/" xmlns:xs="http://www.w3.org/2001/XMLSchema" version="1.0" targetNamespace="http://vaccination.org/">
    <xs:element name="Exception" type="tns:Exception" />
    <xs:element name="FindHistory" type="tns:FindHistory" />
    <xs:element name="FindHistoryResponse" type="tns:FindHistoryResponse" />
    <xs:element name="UpdateHistory" type="tns:UpdateHistory" />
    <xs:element name="UpdateHistoryResponse" type="tns:UpdateHistoryResponse" />
    <xs:complexType name="UpdateHistory">
        <xs:sequence>
            <xs:element name="arg0" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="UpdateHistoryResponse">
        <xs:sequence>
            <xs:element name="return" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="Exception">
        <xs:sequence>
            <xs:element name="message" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="FindHistory">
        <xs:sequence>
            <xs:element name="arg0" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="FindHistoryResponse">
        <xs:sequence>
            <xs:element name="return" type="xs:string" minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```