Patient Identifier Cross-reference Consumer

Architecture & API Documentation

Version 0.2.0

srrenly{at}us{dot}ibm{dot}com | Sondra R Renly

Contents

1. Introduction

The Eclipse Foundation is a not-for-profit corporation formed to advance the creation, evolution, promotion, and support of the Eclipse Platform and to cultivate both an open source community and an ecosystem of complementary products, capabilities, and services. Eclipse is an open source community whose projects are focused on providing an extensible development platform and application frameworks for building software.

℞ www.eclipse.org

The Eclipse Open Healthcare Framework (Eclipse OHF) is a project within Eclipse formed for the purpose of expediting healthcare informatics technology. The project is composed of extensible frameworks and tools which emphasize the use of existing and emerging standards in order to encourage interoperable open source infrastructure, thereby lowering integration barriers.

℞ www.eclipse.org/ohf

The Integrating the Healthcare Enterprise (IHE) is an initiative by healthcare professionals and industry to improve the way computer systems in healthcare share information. IHE promotes the coordinated use of established standards such as DICOM and HL7 to address specific clinical needs in support of optimal patient care. Systems developed in accordance with IHE communicate with one another better, are easier to implement, and enable care providers to use information more effectively.

℞ www.ihe.net

The IHE Technical Frameworks are a resource for users, developers and implementers of healthcare imaging and information systems. They define specific implementations of established standards to achieve effective systems integration, facilitate appropriate sharing of medical information and support optimal patient care. They are expanded annually, after a period of public review, and maintained regularly by the IHE Technical Committees through the identification and correction of errata.

℞ http://www.ihe.net/Technical_Framework/index.cfm

This document corresponds to the Eclipse OHF plugin implementation of the IHE ITI Technical Framework actor Patient Identifier Cross-reference Consumer for the implementation of the ITI-9 PIX Query Transaction that was used successfully by several OHF adopters at the 2007 IHE Connectathon.

2. Getting Started

2.1 Platform Requirements

Verify that the following platform requirements are installed on your workstation, and if not follow the links provided to download and install.

Eclipse SDK 3.2   http://www.eclipse.org/downloads/

Java JDK 1.4.2   http://java.sun.com/javase/downloads/index.jsp

Note that in early 2007, all OHF components will be allowed to use Java 1.5.

2.2 Source Files

Information on how to access the Eclipse CVS technology repository is found on the eclipse wiki:

http://wiki.eclipse.org/index.php/CVS_Howto

Download from dev.eclipse.org/technology/org.eclipse.ohf/plugins:

- org.eclipse.ohf.ihe.common.hl7v2.client
- org.eclipse.ohf.ihe.pix.consumer

For details regarding plugin contents, see the README.txt located in the resources/doc folder of each plugin.

2.3 Dependencies

The Patient Identifier Cross-reference Consumer has dependencies on other OHF plugins and external sources.

2.3.1 Other OHF Plugins

Patient Identifier Cross-reference Consumer plugins are dependent on additional org.eclipse.ohf project plugins. You also need to check-out the following:

- org.eclipse.ohf.hl7v2.core   HL7v2 message object plugins and dependencies
  org.eclipse.ohf.utilities
  org.apache.commons
  org.apache.axis
  org.xmlpull.v1

- org.eclipse.ohf.ihe.common.mllp   Minimum Lower Level Protocol

- org.eclipse.ohf.ihe.atna.audit   Auditing for messages sent and responses received
  org.eclipse.ohf.ihe.atna.agent
  org.eclipse.ohf.ihe.atna.transport
  org.eclipse.ohf.ihe.common.atna

- org.apache.log4j   Debug, warning, and error logging

2.3.2 External Sources

Message defaults and message restrictions can be specified in an optional XML conformance profile. A sample Q23 Get Corresponding IDs (HL7v2.5) conformance profile is included with this plugin in the /resources/conf folder.

2.4 Resources

The following resources are recommended.

2.4.1 Other OHF plugin documentation

The following OHF plugin documents are related to the Patient Identifier Cross-reference Consumer:

- OHF ATNA Audit Client (http://wiki.eclipse.org/index.php/OHF_IHE_Client_plugins)

2.4.2 HL7 Standard 2.5

The Patient Identifier Cross-reference Consumer references standards HL7 version 2.5.

http://www.hl7.org.

2.4.3 IHE ITI Technical Framework

Nine IHE IT Infrastructure Integration Profiles are specified as Final Text in the Version 2.0 ITI Technical Framework: Cross-Enterprise Document Sharing (XDS), Patient Identifier Cross-Referencing (PIX), Patient Demographics Query (PDQ), Audit trail and Node Authentication (ATNA), Consistent Time (CT), Enterprise User Authentication (EUA), Retrieve Information for Display (RID), Patient Synchronized Applications (PSA), and Personnel White Pages (PWP).

The IHE ITI Technical Framework can be found on the following website: http://www.ihe.net/Technical_Framework/index.cfm#IT.

2.4.4 Newsgroup

Any unanswered technical questions may be posted to Eclipse OHF newsgroup. The newsgroup is located at news://news.eclipse.org/eclipse.technology.ohf.

You can request a password at: http://www.eclipse.org/newsgroups/main.html.

3. API Documentation

The Patient Identifier Cross-reference Consumer client supports three formats for input. The client will accept:

- a raw HL7 message (String)
- an HL7v2 message object (org.eclipse.ohf.hl7v2.core Message)
- a ITI-9 PIX Query message supporting the message construction of event:

>   QBP^Q23 – Get Corresponding Identifiers

Examples for the three types of inputs are found in the org.eclipse.ohf.ihe.pix.consumer plugin.

org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.tests > HL7PixQuery.java
org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.tests > MSGPixQuery.java
org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.tests > ClientPixQuery.java

The files in src_tests use a TestConfiguration.java file for extracting the various file locations and MLLP connection parameters. Update this file with your settings before running the sample code.

A raw HL7 message string should be used as input when the originating application is fully capable of sending and receiving HL7 messages. In this case, the Patient Identifier Cross-reference Consumer client is simply providing auditing, communication with the PIX server, and optional message verification. Server responses are returned to the caller as raw HL7v2 message strings. (HL7PixQuery)

A message object should be used as input when the originating application is directly using the OHF HL7v2 component which the Patient Identifier Cross-reference Consumer client sits on top of. In this case, the application has taken full responsibility for message creation and reading the response. The Patient Identifier Cross-reference Consumer client is simply providing conversion to raw HL7, auditing, communication with the PIX server, and optional message verification. Server responses are returned to the caller as HL7v2 message objects. (MSGPixQuery)

A ITI-9 PIX Query message should be used as input when the originating application has neither support for raw HL7 nor message objects. The Patient Identifier Cross-reference Consumer client provides a friendly interface to set and read message fields as well as auditing, communication with the PIX server, and optional message verification. Server responses are returned to the caller as PixConsumerResponse objects. (ClientPixQuery)

ITI-9 PIX Query Message Class

>   PixConsumerQuery

ITI-9 PIX Query Server Response Class

>   PixConsumerResponse

3.1 Use Case - ITI-9 PIX Query
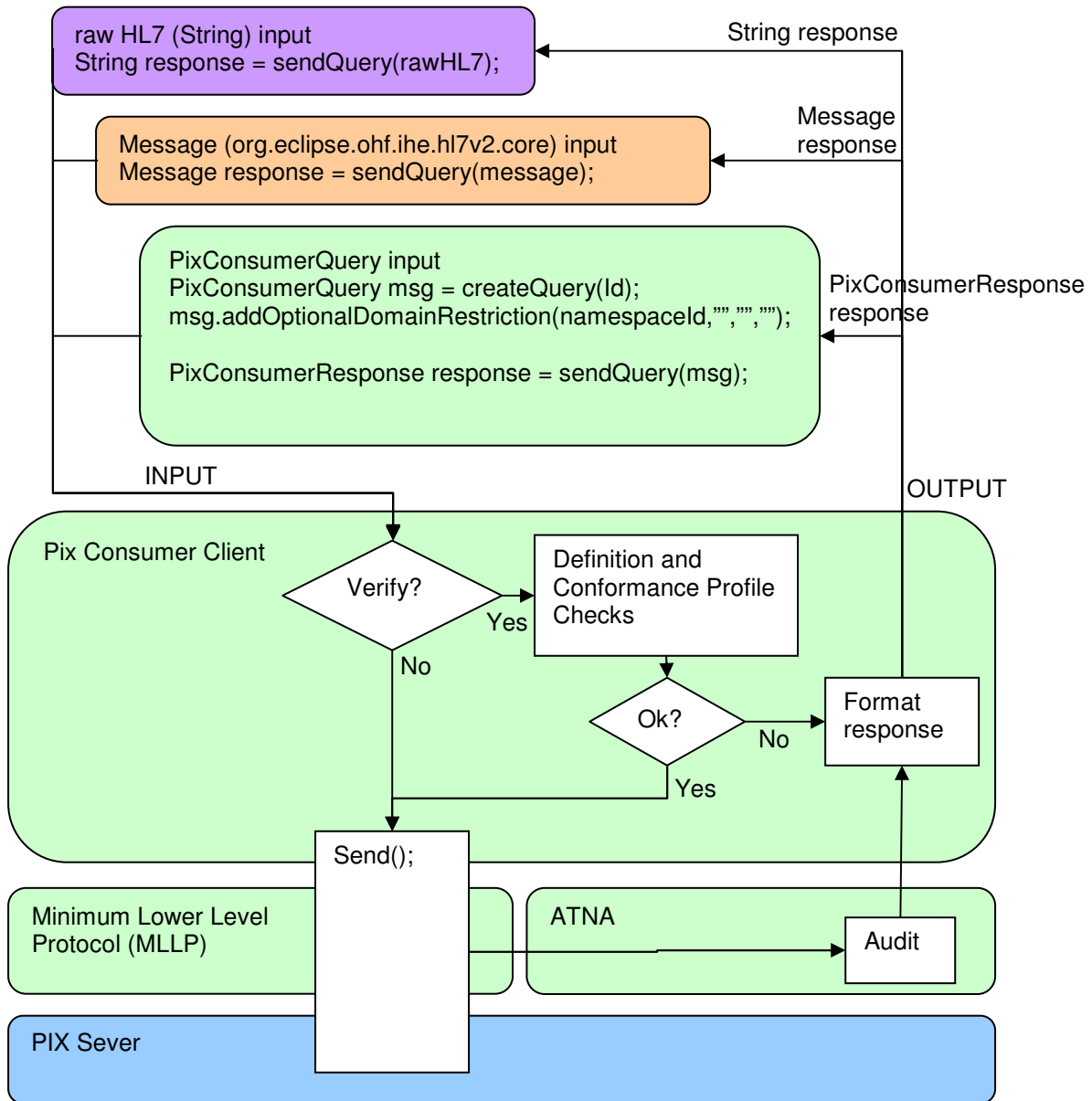
The Patient Identifier Cross-reference Consumer has one transaction. This use case demonstrates in step-by-step and with sample code the creation and use of the Patient Identifier Cross-reference Consumer Client, including the three input options. It includes example client construction of the event specific message object as input but not the creation of raw HL7 or an HL7v2 Message object.

3.1.1 Flow of Execution

raw HL7 (String) input
String response = sendQuery(rawHL7);

String response

Message (org.eclipse.ohf.ihe.hl7v2.core) input
Message response = sendQuery(message);

Message response

PixConsumerQuery input
PixConsumerQuery msg = createQuery(Id);
msg.addOptionalDomainRestriction(namespaceId,"","","");

PixConsumerResponse response = sendQuery(msg);

PixConsumerResponse response

INPUT

OUTPUT

Pix Consumer Client

Verify?

Definition and Conformance Profile Checks

Yes

No

Ok?

Yes

No

Format response

Send();

Minimum Lower Level Protocol (MLLP)

ATNA

Audit

PIX Sever

Create a Patient Identifier Cross-reference Consumer object:

1. Construct ITI-9 PIX Query

2. Construct and associate MLLP (Minimum Lower Level Protocol) Destination

3. Configure auditing.

4. Override the maximum level of validation error allowed before message submission is halted. The levels of error are constants in the OHF HL7v2 CPValidator.java file. The default is to allow up to the warning level.

Create a tailored HL7v2 message object:

1. Create Patient Identifier Cross-reference Consumer Message. Message field defaults are obtained first from the associated Conformance Profile. Required fields not found there default to settings for the IBM Test Dallas Server. (http://ibmod235.dal-ebis.ihost.com:9080/IBMIHII/serverInfoOHF.htm)

2. Override default settings from conformance profile.

3. Add optional field values.

4. As not all fields have a corresponding method, use the generic method to set these additional values. Use method .setField(path, data).

The Patient Identifier Cross-reference Consumer supports populating data in MSH, QPD, and RCP segments. Information about the fields, components, and sub-components available in these segments is available in the HL7 Version 2.5 Standard document in Chapter 2 Section 2.15 Message Control Segments (MSH) and Chapter 5 Section 5.5 Query/Response Message Segments (QPD/RCP). Note that the ITI-9 PIX Query RCP segment is complete when the message is created and no further manipulation is required.

Send the message:

1. Send message

Read the response message:

1. Read response message fields.

3.1.2 Source Code

Create a Patient Identifier Cross-reference Consumer Query:

1. Construct ITI-9 PIX Query

   There are two ways to construct the ITI-9 PIX Query client. At client creation, HL7 definitions are now automatically loaded for you from the HL7v2 toolkit. You can optionally provide an XML conformance profile for providing message defaults and additional message validation restrictions. The conformance profile can be added at a later time as well.

   In this sample code, TConfig refers to the TestConfiguration.java file mentioned in the beginning of this section. See this file for example formatting of these constants.

   ```
   //pixQuery set-up
   PixConsumer pixQuery = new PixConsumer();
   ```

```
//pixQuery set-up with conformance profile
InputStream cpStream = new FileInputStream(TConfig.CPROFILE_PATH);
PixConsumer pixQuery = new PixConsumer(cpStream);
cpStream.close();
```

2. Construct and associate MLLP (Minimum Lower Level Protocol) Destination

Un-Secure Connection:
```
MLLPDestination mllp = new MLLPDestination(TConfig.MLLP_URI);
MLLPDestination.setUseATNA(true);
pixQuery.setMLLPDestination(mllp);
```

Secure Connection:
```
MLLPDestination mllps = new MLLPDestination(TConfig.MLLPS_URI);
MLLPDestination.setUseATNA(true);
pixQuery.setMLLPDestination(mllps);

Properties props = new Properties();
props.setProperty(SecurityDomain.JAVAX_NET_SSL_KEYSTORE, "/x.jks");
props.setProperty(SecurityDomain.JAVAX_NET_SSL_KEYSTORE_PASSWORD, "pswd");
props.setProperty(SecurityDomain.JAVAX_NET_SSL_TRUSTSTORE, "/y.jks");
props.setProperty(SecurityDomain.JAVAX_NET_SSL_TRUSTSTORE_PASSWORD, "pswd");
SecurityDomain domain = new SecurityDomain("domainXY", props);
ConfigurationManager.registerDefaultSecurityDomain(domain);
```

3. Configure auditing.

```
String auditUser = "Bob Smith";
AtnaAgentFactory.getAtnaAgent().setInitiatingUser(auditUser);
AtnaAgentFactory.getAtnaAgent().setAuditRepository(TConfig.ATNA_URI);
```

4. Override the maximum level of validation error allowed before message submission is halted. The levels of error are constants in the OHF HL7v2 CPValidator.java file. The default is to allow up to the warning level.

```
ITEM_TYPE_INFORMATION = 1;
ITEM_TYPE_WARNING = 2;        <<< Default
ITEM_TYPE_ERROR = 3;
ITEM_TYPE_FATAL = 4;

pixQuery.setMaxVerifyEvent(CPValidator.ITEM_TYPE_INFORMATION);
```

Create a tailored HL7v2 message object:

1. Create Patient Identifier Cross-reference Consumer Message.

   It is highly recommended to use the Patient Identifier Cross-reference Consumer to create the query rather than use the PixConsumerQuery constructor as the underlying manager is taken care of. The patient ID parameter is the QPD-3-1 field and the assigningAuthority parameters are within the QPD-3-4 field.

```
PixConsumerQuery msg = pixQuery.createQuery(patientId);
or
PixConsumerQuery msg = pixQuery.createQuery(patientId,
                    namespaceId, universalId, universalIdType);
```

2. Override default settings from conformance profile.

```
msg.changeDefaultCharacterSet("UNICODE");
```

3. Add optional query values.
   You may add multiple domain restrictions by calling this method for each individual domain.

```
msg.addOptionalDomainRestriction("OHF", "", "");
```

4. As not all fields have a corresponding method, use the generic method to set these additional values.
   Use method .setField(alias, data).

```
msg.setField("MSH-12-1", version);
```

   Note that this method does not handle repetitions, such as required by the Domain Restriction QPD-4-4 field. If the additional values require repeat support, use the methods exposed through the Eclipse OHF HL7v2 Message object. See the PixConsumerQuery addOptionalDomainRestriction method for sample code.

Send the message:

1. Send message

```
String response = pixQuery.sendQuery(msg, isValidateOn);
Message response = pixQuery.sendQuery(msg, isValidateOn);
PixConsumerResponse
response = pixQuery.sendQuery(msg, isValidateOn);
```


Read the response message:

1. Read response

```
//HL7v2 message object
msg.getElement("MSA-1").getAsTableCode();        //message AckCode
msg.getElement("QAK-2").getAsTableDescription();  //message QueryStatus

//PixConsumerResponse object
response.getResponseAckCode(isExpandCodeToString); //message AckCode
response.getQueryStatus(isExpandCodeToString);     //message QueryStatus
```

4. Security

4.1 Node Authentication

Transport Layer Security Protocol (TLS) is supported by creating a secure MLLP connection. Information required to instantiate a secure connection to one of the IBM Dallas Servers is available in the sample code configuration file. For more information and use terms on the IBM Dallas Servers, see

 http://lswin10.dfw.ibm.com:9080/IBMIHII/serverInfoIHE_ConnectathonHIMSS2007.htm

4.2 Auditing

Auditing to an Audit Record Repository is automatically enabled through the ATNA Agent. The Patient Identifier Cross-reference Consumer automatically generates the following audit messages:

- EventID 110100 - Actor Start audit message (EventTypeCode 110120)
- EventID 110112 - Query audit message
- EventID 110107 - Import audit message
- EventID 110100 - Actor Stop audit message (EventTypeCode 110121)

5. Configuration

There are two types of configuration in this release.

5.1 Conformance Profile

Create message default field values, such as message header fields, can now be read from the conformance profile field ConstantValue attribute. This is now supported at all levels: field, component, and sub-component.

Field example:

```
<Field Name="MSH-1 Field Separator" Usage="R" Min="1" Max="1" Datatype="ST"
Length="1" ItemNo="00001" ConstantValue="|"></Field>
```

Component example (in this case only the namespaceId is defaulted):

```
<Field Name="MSH-3 Sending Application" Usage="R" Min="0" Max="1" Datatype="HD"
Length="227" Table="0361" ItemNo="00003">

     <Component Name="MSH-3-1 sending application: namespace ID" Usage="R"
     Datatype="IS" ConstantValue="OHFConsumer1"></Component>

     <Component Name="MSH-3-2 sending application: universal ID" Usage="O"
     Datatype="ST"></Component>

     <Component Name="MSH-3-3 sending application: universal ID type" Usage="O"
     Datatype="ID"></Component>

</Field>
```

Sub-component example (default the local domain):

```
<Field Name="QPD-3 Person Identifier" Usage="R" Min="1" Max="1" Datatype="CX"
Length="250">

     <Component Name="QPD-3-4 person identifier: assigning authority" Usage="O"
     Datatype="HD">

          <SubComponent Name="QPD-3-4-1 person identifier: assigning
          authority: namespace ID" Usage="O" Datatype="IS"
          ConstantValue="XREF2005"></SubComponent>

          <SubComponent Name="QPD-3-4-2 person identifier: assigning
          authority: universal ID" Usage="O" Datatype="ST"
          ConstantValue="1.3.6.1.4.1.21367.2005.1.2"></SubComponent>

          <SubComponent Name="QPD-3-4-3 person identifier: assigning
          authority: universal ID type" Usage="O" Datatype="ID"
          ConstantValue="ISO"></SubComponent>

     </Component>
</Field>
```

5.2 Test Configuration

The sample code files in src_tests use a TestConfiguration.java file for extracting the various file locations and MLLP connection parameters. Update the default values in this file as needed with your settings before running the code. Here are the fields that are configured in this file:

```
//basics
public static final String DATA_PATH
public static final String LOG4J_PATH

//HL7PixQuery - run from file
public static final String HL7FILE_PATH

//Conformance profile for second level HL7 verification and defaults
public static final String CPROFILE_PATH

//MLLP Connectivity:
public static URI MLLP_URI
public static URI MLLPS_URI

//TLS: Secure connection parameters
public static final String MLLP_KEYSTORE_NAME
public static final String MLLP_KEYSTORE_PASSWORD
public static final String MLLP_TRUSTSTORE_NAME
public static final String MLLP_TRUSTSTORE_PASSWORD
```

6. Debugging Recommendations

Log statements have been entered throughout the Patient Identifier Cross-reference Consumer plugin source code for assistance in monitoring the progress of the running client. To enable logging, there is a Log4j configuration file.

An example log4j configuration file is in the file /resources/conf/pixconsumer_log4j.xml. The default configuration creates the log file in the folder /resources/log.

7. IHE Connectathon MESA Tests

For current information, please refer to the wiki:
http://wiki.eclipse.org/OHF_MESA_Test_Status

7.1 Plugin Testing

The OHF Patient Identifier Cross-reference Consumer plugin completed testing with the following junits and test scripts.

org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.test.mesa >

MESA10501Test.java	Pix Query Case 1 and Case 2
MESA10502Test.java	Pix Query Case 3
MESA10503Test.java	Pix Query Case 4
MESA2007.txt	Sample script for starting the mesa server and running all tests.

7.2 Bridge Testing

The OHF Bridge completed testing with the following junits and test scripts.

org.eclipse.ohf.bridge.ws > src_tests > org.eclipse.ohf.bridge.ws.tests.mesa >

TestMESA10501a.java	Pix Query Case 1 and Case 2
TestMESA10501b.java	Pix Query Case 1 and Case 2
TestMESA10502.java	Pix Query Case 3
TestMESA10503.java	Pix Query Case 4

org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.test.mesa >

MESA2007.txt	Sample script for starting the mesa server and running all tests.

8. IHE Connectathon Tests

8.1 Plugin Testing

The OHF Patient Identifier Cross-reference Consumer plugin completed testing with the following junit.

org.eclipse.ohf.ihe.pix.consumer > src_tests > org.eclipse.ohf.ihe.pix.consumer.test.connectathon >

ConnectathonFarnsworth.java   Pix Query on Steve Farnsworth