Patient Identity Source

Architecture & API Documentation

Version 0.0.1

srrenly@us.ibm.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 (EOHF) 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 documentation addresses the alpha release of the Eclipse OHF plugin implementation of the IHE ITI Technical Framework actor Patient Identity Source for the implementation of the ITI-8 Patient Identity Feed Transaction.

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 5.0	http://java.sun.com/javase/downloads/index.jsp

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.source

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

2.3 Dependencies

The Patient Identity Source has dependencies on both other OHF plugins and external sources.

2.3.1 Other OHF Plugins

Patient Identity Source plugins are dependent on additional org.eclipse.ohf project plugins. You also need to check-out the following:

• org.eclipse.ohf.hl7v2.core org.eclipse.ohf.utilities org.apache.axis org.xmlpull.v1	HL7v2 message object plugins and dependencies
• 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.common.hl7v2	HL7v2 segment, field definitions (temporary)
• org.apache.log4j	Debug, warning, and error logging

2.3.2 External Sources

The HL7v2 plugins currently requires a licensed copy of the HL7 access database for the purpose of message object creation and verification. The .mdb file must be placed in the client plugin resources folder under the conf folder.

org.eclipse.ohf.ihe.common.hl7v2.client > resources > conf > hl7_58.mdb

If you have not yet obtained a copy, refer to http://www.hl7.org.

2.4 Resources

The following resources are recommended.

2.4.1 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.

2.4.2 HL7 Standard 2.3.1

The Patient Identity Source references standards HL7 version 2.3.1.

http://www.hl7.org.

2.4.3 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 Identity Source client supports three formats for input. The client will accept:

- a raw HL7 message
- an HL7v2 message object
- an ITI-8 Patient Identity Feed message supporting the manual HL7v2 message construction of:

ADT_A01 – Admission of an in-patient into a facility
ADT_A04 – Registration of an outpatient for a visit of the facility
ADT_A05 – Pre-admission of an in-patient
ADT_A08 – Update patient information
ADT_A40 – Patient Merge – Internal ID

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

org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > HL7PixFeed.java
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > MSGPixFeed.java
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > OtherPixFeed.java

3.1 Creating a Patient Identity Source Object

3.1.1 Flow of Execution

The steps necessary to create a Patient Identity Source object:

1. Construct ITI-8 Patient Identity Feed

```
try {
      pixFeed = new PixSource();
} catch (ClientException e) {
      throw new PixSourceException(e);
}
```

2. Construct MLLP (minimum lower level protocol) Destination

```
mllp = new MLLPDestination(host, port, beginChars, endChars, buffer_size);
```

3. Associate MLLP to ITI-8 Patient Identity Feed

```
pixFeed.setMLLPDestination(mllp);
```

3.1.2 API Details

Constructor Summary
`PixSource()` Constructs a PIX Source message manager object.

Method Summary	
java.lang.String	**getAuditUser**() Get the message audit user.
int	**getMaxVerifyEvent**() Maximum error the message verification allows before submission is blocked.
org.eclipse.ohf.hl7v2.core.message.MessageManager	**getMessageManager**()
org.eclipse.ohf.ihe.common.mllp.MLLPDestination	**getMLLPDestination**() Returns the MLLP destination with TCP settings.
boolean	**isDoAudit**() Returns the doAudit boolean flag.
void	**setAuditUser**(java.lang.String audituser) Set the user to associate with the message.
void	**setDoAudit**(boolean doAudit) Set the doAudit boolean flag.
void	**setMaxVerifyEvent**(int maxVerifyEvent) Maximum error the message verification allows before submission is blocked.
void	**setMessageManager**(MessageManager globalFactory)
void	**setMLLPDestination**(org.eclipse.ohf.ihe.common.mllp.MLLPDestination MLLP) Set the MLLP destination with TCP settings.

3.2 Creating a ITI-8 Patient Identity Feed Message Object

In the case that your source application is neither capable of creating/receiving raw HL7v2 messages nor creating/receiving HL7v2 message objects, you may use this client to create/receive tailored HL7v2 message objects with a friendly interface for setting and reading the field values.

The following HL7 message types are supported:

ADT_A01 – Admission of an in-patient into a facility
ADT_A04 – Registration of an outpatient for a visit of the facility
ADT_A05 – Pre-admission of an in-patient
ADT_A08 – Update patient information
ADT_A40 – Patient Merge – Internal ID

3.2.1 Flow of Execution

The steps necessary to create a tailored HL7v2 message object:

1. Create Patient Identity Source Message Desired
   ```
   PixMsgAdmitInpatient admit = pixFeed.admitInpatient("[patientID]");
   ```
2. Change default settings
   ```
   admit.changeDefaultCharacterSet("UNICODE");
   ```
3. Add optional field values
   ```
   admit.addOptionalAddressStateOrProvince("CA");
   ```
4. If method does not already exist to modify message, use method .setField(field, value).
   ```
   admit.setField("PID-11-1", "123 San Jose Drive");
   ```

The Patient Identity Source supports populating data from MSH, EVN, PID, MRG (if merge message), and PV1 segments. Information about the fields, components, and sub-components available in these segments is available in the HL7 Version 2.3.1 Standard document in Appendix B.

3.2.2 API Details

Method Summary – Create Message	
`PixMsgAdmitInpatient`	**`admitInpatient`**`(java.lang.String patient_id)` ADT_A01 Admit Inpatient - create message
`PixMsgMergePatient`	**`mergePatient`**`(java.lang.String patient_id, java.lang.String patient_class, java.lang.String prior_id)` ADT_A40 Merge Patient - create a message
`PixMsgPreadmitInpatient`	**`preadmitInpatient`**`(java.lang.String patient_id)` ADT_A05 Preadmit Inpatient - create message
`PixMsgRegisterOutpatient`	**`registerOutpatient`**`(java.lang.String patient_id)` ADT_A04 Register Outpatient - create message
`PixMsgUpdatePatient`	**`updatePatient`**`(java.lang.String patient_id, java.lang.String patient_class)` ADT_A08 Update Patient - create message

Method Summary – Change Default Settings	
`void`	**`changeDefaultCharacterSet`**`(java.lang.String charSet)` Character set used to construct this message.

void	**changeDefaultControlID**(java.lang.String control_id) Unique ID used to link the query message to the response message.
void	**changeDefaultProcessEnvironment**(java.lang.String environment) Environment type from which this message originates.
void	**changeDefaultReceivingApplication**(java.lang.String receivingApplication) The unique identifier for the receiving application.
void	**changeDefaultRecievingFacility**(java.lang.String receivingFacility) The unique identifier for the receiving facility.
void	**changeDefaultSendingApplication**(java.lang.String sendingApplication) The unique identifier for the sending application.
void	**changeDefaultSendingFacility**(java.lang.String sendingFacility) The unique identifier for the sending facility.

Method Summary – Add Optional Fields

void	**addOptionalAddressCity**(java.lang.String city) PID-11-3 Patient Address - City
void	**addOptionalAddressCountry**(java.lang.String country) PID-11-6 Patient Address - Country
void	**addOptionalAddressCountyOrParish**(java.lang.String countyOrParish) PID-11-9 Patient Address - County or Parish Code
void	**addOptionalAddressOtherDesignation**(java.lang.String address) PID-11-2 Patient Address - Other Designation
void	**addOptionalAddressStateOrProvince**(java.lang.String stateOrProvince) PID-11-4 Patient Address - State or Province
void	**addOptionalAddressStreet**(java.lang.String address) PID-11-1 Patient Address - Street Address

void	**addOptionalAddressType**(java.lang.String type) PID-11-7 Patient Address - Address Type
void	**addOptionalAddressZipOrPostalCode**(java.lang.String zipOrPostalCode) PID-11-5 Patient Address - Zip or Postal Code
void	**addOptionalDateOfBirth**(java.lang.String dob) PID-7 Date/Time of Birth
void	**addOptionalPatientID**(java.lang.String id_number, java.lang.String assigningAuthorityName, java.lang.String universalID, java.lang.String universalIDType) PID-3 Patient ID (internal)
void	**addOptionalPatientNameFirst**(java.lang.String firstName) PID-5-2 Patient Name - first name
void	**addOptionalPatientNameLast**(java.lang.String lastName) PID-5-1 Patient Name - last name
void	**addOptionalPatientNameMiddle**(java.lang.String middleName) PID-5-3 Patient Name - middle name
void	**addOptionalPatientNamePrefix**(java.lang.String prefix) PID-5-6 Patient Name - prefix
void	**addOptionalPatientNameSuffix**(java.lang.String suffix) PID-5-4 Patient Name - suffix
void	**addOptionalPatientNameTitle**(java.lang.String title) PID-5-5 Patient Name - title
void	**addOptionalPhoneBusiness**(java.lang.String businessPhone) PID-14 Business Phone
void	**addOptionalPhoneHome**(java.lang.String homePhone) PID-13 Home Phone
void	**addOptionalSex**(java.lang.String sex) PID-8 Administrative Sex
void	**setField**(java.lang.String alias, java.lang.String data) Updates message object structure with data.

3.3 Sending the ITI-8 Patient Identity Feed Message

3.3.1 Flow of Execution

The steps necessary to send the message:

1. Send message

```
response = pixFeed.sendHL7(admit, verify);
response = pixFeed.sendMsg(admit, verify);
response = pixFeed.sendAdmission(admit, verify);
```

3.3.2 API Details

Method Summary – Send Message	
`PixSourceResponse`	**`sendAdmission`**`(PixMsgAdmitInpatient msg, boolean verify)` ADT_A01 Admit Inpatient - send message
`PixSourceResponse`	**`sendMerge`**`(PixMsgMergePatient msg, boolean verify)` ADT_A40 Merge Patient - send message
`PixSourceResponse`	**`sendPreAdmission`**`(PixMsgPreadmitInpatient msg, boolean verify)` ADT_A05 Preadmit Inpatient - send message
`PixSourceResponse`	**`sendRegistration`**`(PixMsgRegisterOutpatient msg, boolean verify)` ADT_A04 Register Outpatient - send message
`PixSourceResponse`	**`sendUpdate`**`(PixMsgUpdatePatient msg, boolean verify)` ADT_A08 Update Patient - send message
`java.lang.String`	**`sendHL7`**`(java.lang.String rawHL7, boolean verify)` Processes HL7 messages with optional intermediate verification.
`org.eclipse.ohf.hl7v2.core.message.model.Message`	**`sendMsg`**`(org.eclipse.ohf.hl7v2.core.message.model.Message msg, boolean verify)` Process Message Object message with optional intermediate verification.

3.4 Reading a ITI-8 Patient Identity Feed Response Message

3.4.1 Flow of Execution

The steps necessary to create a tailored HL7v2 message object:

1. Read Response

```
response.getResponseAck(true);
response.getControlID();
response.getErrorCodeAndLocation();
```

3.4.2 API Details

Method Summary	
java.lang.String	**getCharacterSet**() MSH-18 Character Set
java.lang.String	**getControlID**() MSA-2 Message Control ID
java.lang.String	**getErrorCodeAndLocation**() ERR-1 HL7 Error Code and Location
java.lang.String	**getProcessEnvironment**(boolean expandString) MSH-11 Processing ID
java.lang.String	**getReceivingApplication**(java.lang.String receivingApplication) MSH-5 Receiving Application
java.lang.String	**getReceivingFacility**() MSH-6 Receiving Facility
java.lang.String	**getResponseAck**(boolean expandString) MSA-1 Acknowledgement Code
java.lang.String	**getSendingApplication**() MSH-3 Sending Application
java.lang.String	**getSendingFacility**() MSH-4 Sending Facility

4. Sample Code

For example implementations, see

org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > HL7PixFeed.java
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > MSGPixFeed.java
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > OtherPixFeed.java

4.1 Raw HL7

In the happy circumstance that your source application is fully capable of creating/receiving raw HL7v2 messages, you may use this client as a middle-layer to verify, audit, and communicate with the PIX/PDQ server. Server responses are returned to the caller as raw HL7v2 message strings.

For example implementation, see
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > HL7PixFeed.java

4.2 HL7v2 Message Object

In the happy circumstance that our source application if capable of creating/receiving HL7v2 message objects, you may use this client as a middle-layer to verify, convert to raw HL7, audit, and communicate with the PIX/PDQ server. Server responses are returned to the caller as HL7v2 message objects.

For example implementation, see
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > MSGPixFeed.java

4.3 ITI-8 Patient Identity Feed Message Object

In the case that your source application is neither capable of creating/receiving raw HL7v2 messages nor creating/receiving HL7v2 message objects, you may use this client to create/receive tailored HL7v2 message objects with a friendly interface for setting and reading the field values.

ITI-8 Patient Identity Feed Message Classes

PixMsgAdmitPatient
PixMsgRegisterOutpatient
PixMsgPreadmitInpatient
PixMsgUpdatePatient
PixMsgMergePatient

ITI-8 Patient Identity Feed Server Response Class

PixSourceResponse

For example implementation, see
org.eclipse.ohf.ihe.pix.source > src_tests > org.eclipse.ohf.ihe.pix.source.tests > OtherPixFeed.java