Open Healthcare Framework Bridge

Architecture & API Documentation

mattadav@us.ibm.com | Matthew Davis

kxjiang@us.ibm.com | Kelvin Jiang

monvura@us.ibm.com | Melih Onvural

ioprenc@us.ibm.com | Ivan Oprencak

Contents

1. Introduction

The United States Government is committed to apply modern information technology to private clinics and public hospitals nationwide by 2014. In 2006, President George W. Bush listed five key policies to modernize healthcare technologies, one of which was to implement, “a nationwide information network [that] will protect the privacy of a patient's medical information while making health information available in real-time.”

The Eclipse Foundation, under its Technology Projects initiative, is dedicated to design a common healthcare framework that will interconnect clinics, hospitals, and labs both on a regional level and on a national level to heed the call to build an integrated healthcare infrastructure. The Eclipse Open Healthcare Foundation (OHF) is building IHE standards-compliant plug-ins to enable end-to-end interoperability between healthcare applications. While the ultimate goal is to build RCP applications on top of the Eclipse development framework, the current status of healthcare applications calls for interoperability through existing systems. To support existing systems, the OHF project will develop the OHF Bridge.

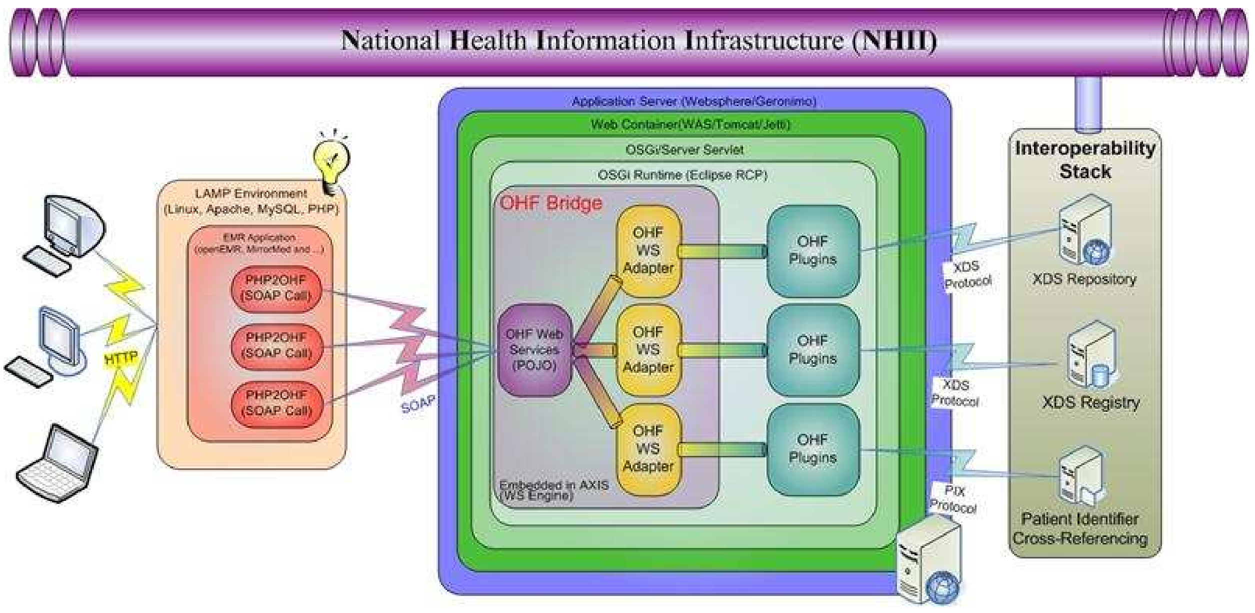
The OHF Bridge component of the OHF project provides existing healthcare applications with a platform-independent plug-in that defines patient information search and retrieval methods to and from a registry and repository. It exposes the necessary methods through a single web service, and then, after formatting the incoming data, passes the data through to the OHF plug-ins. Healthcare applications interacting with the OHF Bridge may include electronic medical records (EMR), personal health records (PHR), radiology imaging systems, lab systems, etc. By exposing the OHF plug-ins to existing healthcare applications in a platform-independent manner, the OHF Bridge hopes to facilitiate the adoption of OHF plug-ins as the means through which healthcare interoperability becomes a reality.

Specifically, this plug-in addresses the need to connect healthcare applications to a backend repository using available web services technologies. Since the transport layer is through web services, any environment that supports web services, such as LAMP, .NET, Java, C++, Python, etc., can use the OHF Bridge plug-in. This documentation will address the design and implementation of this plug-in.

2. System Architecture

2.1 Server Architecture

The OHF Bridge components live inside of an application server. Specifically, they are deployed inside of a Tomcat instance which has another server running inside of it. This second server is an OSGi instance built on the Eclipse Equinox project's implementation of the OSGi framework. Within the OSGi instance, there is an Axis server, exposed as a servlet. This servlet facilitiates the SOAP communication between the healthcare applications using the OHF Bridge and the OHF plug-ins themselves. Explaining the full capabilities of Tomcat, OSGi, or Axis is beyond the scope of this documentation, but each will be introduced in relation to their role in the OHF Bridge. The system architecture is represented in figure 2-1 below.



National Health Information Infrastructure (NHII)
Application Server (Websphere/Geronimo)
Web Container(WAS/Tomcat/Jetti)
OSGi/Server Servlet
OSGi Runtime (Eclipse RCP)
OHF Bridge
LAMP Environment (Linux, Apache, MySQL, PHP)
EMR Application (openEMR, MirrorMed and ...)
PHP2OHF (SOAP Call)
PHP2OHF (SOAP Call)
PHP2OHF (SOAP Call)
HTTP
SOAP
OHF Web Services (POJO)
OHF WS Adapter
OHF WS Adapter
OHF WS Adapter
OHF Plugins
OHF Plugins
OHF Plugins
Embedded in AXIS (WS Engine)
XDS Protocol
XDS Protocol
PIX Protocol
Interoperability Stack
XDS Repository
XDS Registry
Patient Identifier Cross-Referencing

Figure 2-1. – System Architecure showing the full OHF implementation

2.1.2 Tomcat

Tomcat acts as the web container for the OHF Bridge component. It is all that is needed to expose the component properly, but the OHF Bridge component can also be nested inside of an application server. The installation and configuration of Tomcat is beyond the scope of this document, but more help can be found at Tomcat's website (***http://tomcat.apache.org/***).

2.1.2 OSGi on Server

The implementation of an OSGi on Server instance is done through the packages released in the Eclipse Equinox project. The mission of the Equinox project (from there website):

"Equinox is an implementation of the OSGi R4 core framework specification, a set of bundles that implement various optional OSGi services and other infrastructure for running OSGi-based systems"

The OSGi on Server implementation that is being used as the container of OHF Bridge allows for plug-ins to be implemented as bundles and installed, started, and updated during runtime. For the purpose of the OHF Bridge, it's most important to be able to successful implement an Axis plug-in as well as the web service plug-ins that transport the SOAP messages from the 3rd-party healthcare applications to the OHF plug-in components. The other OHF components are also plug-ins that can be installed in the OSGi on Server instance.

2.1.3 Axis Servlet

The handling of SOAP messages is done through Apache Axis. Axis is implemented as a bundle in OSGi on Server. This approach allows it to communicate with the 3rd-party healthcare applications. The objects passed through Axis are then transformed into the correct types to be passed to the OHF plug-ins. Axis returns the result of a query as another SOAP message. Communication outside of the OHF Bridge is conducted through Apache Axis.

2.2 Bundles

The words bundle and plug-in are interchangeable in the OSGi on Server environment in which the OHF Bridge resides. There are two bundles that are important to the Bridge itself. The first is the web services bundle, which exposes its methods to 3rd-party healthcare applications. The second is the bridge bundle which handles the data passed in through the web services and properly transforms them so that they can be used by the OHF plug-ins.

2.2.1 Web Services Bundle

The web services bundle exposes its methods to the public through one class to accept and pass on the data from a 3rd-party healthcare application to the Bridge bundle. It functions solely as an intermediary.

2.2.2 Bridge Bundle

The Bridge bundle transforms the data that it receives from the Web Service bundle into types that can be recognized by the OHF plug-ins that enact the XDS and PIX/PDQ queries. It takes advantage of the Adaptor pattern to perform this task. Once the transformation is complete, the Bridge bundle calls the necessary methods to enact a query. The Bridge bundle handles both PIX/PDQ and XDS queries

3. API Documentation

The following web services are implemented in Java using Apache AXIS and are accessible via a general web service consumer. A WSDL file is provided as a descriptor for these web services and can be used directly for consumption purposes.

Note: The components of the following API are subject to change, including but not limited to web service association, operation names, prototype definitions and data structures. This documentation will be updated periodically to reflect changes in the general web service API.

3.1 Search for Patients

To search patients, the OHF Bridge web service provides a method which takes as a parameter a PatientInfoType object that holds a potential patient's demographic information. The PatientInfoType is passed through SOAP to the OHF plugins where a query is run against the given information. An array of patients who match the given criteria are returned, and the search can be further refined or the sought patient's information can be retrieved.

3.1.1 Methods

SearchPatient

SearchPatient sends over SOAP a local object that is mapped to a PatientInfoType object. The PatientInfoType is populated with the search terms against which the query is happening. The result of the query is returned as an array of PatientInfoType objects.

Request

Parameter	Type	Description	Required?
patientInfo	PatientInfoType	The demographic information upon which the search will be conducted	Yes

Response

SearchPatient returns an array of type PatientInfoType which is the result of the query run against the information passed into the Bridge.

searchPatient

Note: searchPatient is deprecated, but is described here to support legacy applications. searchPatient sends over SOAP a local object that is mapped to a

PDQQueryObject object. The PDQQueryObject is populated with the search terms against which the query is happening. The result of the query is returned as an array of PDQQueryObjects objects.

Request

Parameter	Type	Description	Required?
patientInfo	PDQQueryObject	The demographic information upon which the search will be conducted	Yes

Response

searchPatient returns an array of type PDQQueryObject which is the result of the query run against the information passed into the Bridge.

3.1.2 Object Types

PatientInfoType

An object composed of the demographic objects that describe a patient

Fields

Name	Type	Description	Required?
patientIdentifier	PIDType	The object which stores the four components of the patient identifier	No
patientName	PatientNameType	The object which stores the components that make up a patient's name	No
patientDateOfBirth	String	The date of birth for the patient	No
patientSex	String	The sex of the patient	No
patientAddress	AddressType	The object which stores the components that make up a patient's address	No
patientPhoneHome	PhoneType	The object which stores the patient's home telephone number	No
patientPhoneBusiness	PhoneType	The object which stores the patient's business telephone number	No

PIDType

An object composed of the components necessary to completely describe a patient's identifier

Fields

Name	Type	Description	Required?
idNumber	String	The identifier number of the patient	No
assigningAuthorityName	String	The name of the assigning authority which created the user	No
assigningAuthorityUniversalId	String	The universal identifier for the assigning authority which created the user	No
assigningAuthorityUniversalIdType	String	The type of the universal identifier for the assigning authority which created the user	No

PatientNameType

An object composed of the components that make up a patient's name

Fields

Name	Type	Description	Required?
familyName	String	The patient's family name, also referred to as the patient's last name	No
givenName	String	The patient's given name, also referred to as the patient's first name	No
otherName	String	Any other names by which the patient is known inlcuding middle name, aliases, nicknames, etc.	No
suffix	String	The suffix appended to the patient's name, such as Sr., Jr., III, etc.	No
prefix	String	The prefix appended to the patient's name, such as Miss, Mr., Dr., etc.	No

AddressType

An object composed of the components that make up a patient's address

Fields

Name	Type	Description	Required?
streetAddress	String	The street address component of the patient's address	No
otherDesignation	String	Any other component of the address that isn't otherwise included such as P.O. Box, Apt. No., etc.	No
city	String	The city component of the patient's address	No
stateOrProvince	String	The state/province component of the patient's address	No
zipOrPostalCode	String	The zip/postal code component of the patient's address	No
country	String	The country component of the patient's address	No
countyParishCode	String	The county/parish component of the patient's address	No

PhoneType

An object which stores a telephone number attached to the patient

Fields

Name	Type	Description	Required?
unformattedTelephoneNumber	String	The unformatted telephone number of the patient	No

PDQQueryObject

PDQQueryObject is deprecated, but is still shown here for legacy support. It is a composite object composed of the fields necessary to query against patient information.

Fields

Name	Type	Description	Required?
patientId	String	The patient id by which an individual is stored in the repository	No
lastName	String	A patient's last name	No
firstName	String	A patient's first name	No
middleName	String	A patient's middle name	No

nameSuffix	String	A patient’s suffix in their name (such as Jr., Sr.)	No
namePrefix	String	A patient’s prefix to their name (such as Mrs., Mr., Dr., etc.)	No
nameTitle	String		No
sex	String	The patient’s gender	No
dateOfBirth	String	A patient’s date of birth	No
addressStreet	String	A patient’s street address	No
addressCity	String	A patient’s city address	No
addressCountyOrParish	String	A patient’s county or parish based on the local region in which they live and its terminology	No
addressStateOrProvince	String	A patient’s state or province based on the local region in which they live and its terminology	No
addressCountry	String	A patient’s nation of residence	No
addressZipOrPostalCode	String	A patient’s zip code or postal code based on the local region in which they live and its terminology	No
addressType	String	A descriptor for what address the patient is providing (business, home, P.O. Box, etc.)	No
addressOtherDesignation	String		No
phoneHome	String	A patient’s home telephone number	No
phoneBusiness	String	A patient’s work telephone number	No

3.2 Retrieve Documents

To retrieve documents, the OHF Bridge web service provides a series of methods which find and retrieve the documents associated with a given patient identifier. It’s important to have the patient identifier that the server associates with the patient, and this can be done using the appropriate patient search.

3.2.1 Methods

FindDocumentsByPatientId

FindDocumentsByPatientId searches the XDS registry for a list of documents associated with a given patient identifier. FindDocumentsByPatientId returns an XDSDocCollection which contains an array of XDSDocType objects which contain document metadata and information for further retrieval.

Request

Parameter	Type	Description	Required?
patientId	String	The patient identifier to which the sought documents are attached	Yes

Response

FindDocumentsByPatientId returns an XDSDocCollection object which is populated by an array of XDSDocType objects which contain document metadata or null if no responses were found.

Exceptions

FindDocumentsByPatientId throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

GetDocumentByUniqueId

Queries for and retrieves an XDS document from the repository based on the "unique identifier" assigned to each document placed in the repository. Returns a single object of type XDSDocType that contains document data and metadata.

Request

Parameter	Type	Description	Required?
uniqueId	String	"Unique ID" from which to retrieve the document from the XDS registry.	Yes

Response

Returns a single object of type XDSDocType with the document contents and metadata or null if the document was not found.

Exceptions

GetDocumentByUniqueId throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

GetDocumentByUUID

Queries for and retrieves an XDS document from the repository based on the Universally Unique Identifier (UUID) assigned to each document placed in the registry and repository. Returns a single object of type XDSDocType that contains document data and metadata.

Request

Parameter	Type	Description	Required?
UUID	String	Universally unique identifier (UUID) from which to retrieve the document from the XDS registry.	Yes

Response

Returns a single object of type XDSDocType with the document contents and metadata or null if the document was not found.

Exceptions

GetDocumentByUUID throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

getDocumentByPatientId

Note: getDocumentByPatientId is deprecated, but is described here to support legacy applications. getDocumentByPatientId searches the XDS registry for a list of documents associated with a given patient identifier. getDocumentByPatientId returns an array of XDSBridgeDocument objects which contain document metadata and information for further retrieval.

Request

Parameter	Type	Description	Required?
patientId	String	"Unique ID" from which to retrieve the document from the XDS registry	Yes

Response

getDocumentByPatientId returns an array of XDSBridgeDocument objects which contain document metadata or null if no responses were found.

Exceptions

getDocumentByPatientId throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

searchByPatientId

Note: searchByPatientId is deprecated, but is described here to support legacy applications. searchByPatientId searches the XDS registry for a list of documents associated with a given patient identifier. searchByPatientId returns an array of XDSBridgeDocument objects which contain document metadata and information for further retrieval.

Request

Parameter	Type	Description	Required?
patientId	String	"Unique ID" from which to retrieve the document from the XDS registry	Yes

Response

searchByPatientId returns an array of XDSBridgeDocument objects which contain document metadata or null if no responses were found.

Exceptions

searchByPatientId throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

getDocumentByUniqueId

Note: getDocumentByUniqueId is deprecated, but is described here to support legacy applications. Queries for and retrieves an XDS document from the repository based on the "unique identifier" assigned to each document placed in the repository. Returns a single object of type XDSBridgeDocument that contains document data and metadata.

Request

Parameter	Type	Description	Required?
uniqueId	String	"Unique ID" from which to retrieve the document from the XDS registry.	Yes

Response

Returns a single object of type XDSBridgeDocument with the document contents and metadata or null if the document was not found.

Exceptions

getDocumentByUniqueId throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

getDocumentByUUID

Note: getDocumentByUUID is deprecated, but is described here to support legacy applications. Queries for and retrieves an XDS document from the repository based on the Universally Unique Identifier (UUID) assigned to each document placed in the registry and repository. Returns a single object of type XDSBridgeDocument that contains document data and metadata.

Request

Parameter	Type	Description	Required?
UUID	String	Universally unique identifier (UUID) from which to retrieve the document from the XDS registry.	Yes

Response

Returns a single object of type XDSBridgeDocument with the document contents and metadata or null if the document was not found.

Exceptions

getDocumentByUUID throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

3.2.2 Objects Types

XDSDocType

Simple data object for holding and mapping document data.

Fields

Name	Type	Description	Required?
patientInfoType	PatientInfoType	Stores the complete identifier information for the patient	No
documentTitle	String	The title component of the document	No
effectiveTime	String	The time in which the document entered the system	No
encodedDocument	String	Document data encoded and stored as a String	No
encodingType	String	Encoding type of the document. This can be either base_64 or text	No
mimeType	String	The mime type of the document	No
documentSize	String	The size of the document	No
uniqueId	String	The unique identifier by which the document can be found	No
uuid	String	The universally unique identifier by which the document can be found	No

PatientInfoType

See Section 3.1.2

XDSDocCollection

Simple data object for storing an array of XDSDocType objects.

Fields

Name	Type	Description	Required?
documentTypeArray	XDSDocType[]	An array of XDSDocType objects	No

XDSBridgeDocument

XDSBridgeDocument is deprecated, but is still shown here for legacy support. It is a data object for holding and mapping XDSBridgeDocument data.

Fields

Name	Type	Description	Required?
Document	String	Base64-encoded binary or ASCII contents for the document.	No
documentTitle	String	Title of the document	No
message (*deprecated*)	String	Server message for error handling.	No
mimeType	String	MIME type for the document in transit.	No
Size	Integer	Size in bytes of the document in transit	No
status (deprecated)	Integer	Server status code for document.	No
Uuid	String	Universally unique identifier (UUID) for the document in transit.	Yes
uniqueId	String	"Unique ID" for document in transit.	Yes

3.3 Submit Documents

To submit documents, the OHF Bridge web service provides a method which given a complete patient identifier can submit a document. It's important to have the complete patient identifier that the server associates with the patient, and this can be done using the appropriate patient search.

3.3.1 Methods

SubmitDocument

SubmitDocument takes an XDSDocType object and passes it to the server after wrapping it as a CDA document to be stored in the XDS repository. SubmitDocument returns a String which contains the UUID value of the document now stored in the XDS repository.

Request

Parameter	Type	Description	Required?
doc	XDSDocType	The document and its metadata that are to be added to the XDS repository	Yes

Response

SubmitDocument returns a String that is the UUID value of the document that is now stored in the XDS repository.

Exceptions

SubmitDocument throws SOAP exceptions, to be handled by the consumer, on the event of server errors.

3.3.2 Objects Types

XDSDocType

Simple data object for holding and mapping document data.

Fields

Name	Type	Description	Required?
patientInfoType	PatientInfoType	Stores complete identifier info for the patient	No
documentTitle	String	The title component of the document	No
effectiveTime	String	The time in which the document entered the system	No
encodedDocument	String	Document data encoded and stored as a String	No
encodingType	String	Encoding type of the document. This can be either base_64 or text	No
mimeType	String	The mime type of the document	No
documentSize	String	The size of the document	No
uniqueId	String	The unique identifier by which the document can be found	No
uuid	String	The universally unique identifier by which the document can be found	No

PatientInfoType

See Section 3.1.2

3.4 Cross Referencing Patients

To cross reference patient identifiers between the local identifier and the CAD identifier, the OHF Bridge provides a method which given a local patient identifier returns the fully qualified CAD identifier.

3.4.1 Methods

CrossReferenceByPatientId

CrossReferenceByPatientId takes a String representation of a local identifier and returns a PIDType that contains all of the components of a CAD identifier. The CAD identifier is required to submit documents.

Request

Parameter	Type	Description	Required?
patientId	PIDType	The local patient identifier against which the CAD identifier will be found	Yes
verify	boolean	True if intermediate verification of the sent message is required	Yes

Response

CrossReferenceByPatientId returns a PIDType with the components of the CAD identifier or null if the local identifier could not be mapped.

3.4.2 Objects Types

PIDType

An object composed of the components necessary to completely describe a patient's identifier

Fields

Name	Type	Description	Required?
idNumber	String	The identifier number of the patient	No
assigningAuthorityName	String	The name of the assigning authority which created the user	No
assigningAuthorityUniversalId	String	The universal identifier for the assigning authority which created the	No

		user	
assigningAuthorityUniversalIdType	String	The type of the universal identifier for the assigning authority which created the user	No

4. Technology Documentation

This chapter describes the sample PHP application that demonstrates the OHF Bridge technology.

4.1 OHF Bridge Demo

The following demo is for the initial standalone version of OHF Bridge, utilizing the OHF PIX/PDQ and XDS plug-ins. Some variation may be made in the steps of the demo; however, due to the limited nature of the repository data, irrelevant or bad data may be returned. The steps outlined return relevant, working data at edit time. It's important to note that this demonstration is not meant to represent an EMR application, nor is it able to show full patient demographic search because of the limited data in the repository.

4.1.1 File Locations

The Bridge demo currently resides at

`http://ibmod235.dal-ebis.ihost.com:8080/ohf/demo/index.php`,

The WSDL files currently reside at:

Search Patients

`http://ibmod235.dal-ebis.ihost.com:8090/bridge/services/ohf-pdq-bridge?wsdl`

Retrieve Documents

`http://ibmod235.dal-ebis.ihost.com:8090/bridge/services/ohf-xds-bridge?wsdl`

4.1.2 Demo Steps

Demonstrate Patient Search Functionality

1. Navigate to **`http://ibmod235.dal-ebis.ihost.com:8080/ohf/demo/index.php`**

Demonstrate Document Retrieval Functionality

Note: Variations may be made in the above steps (i.e. you can search by names, other states, etc) but, due to the limited nature of the repository, valid data may not be available.

4.2 Newsgroup Help

Several developers are currently working on OHF Bridge, and they are happy to help others to implement this technology in their EMR applications. If you have any questions about the implementation of OHF Bridge, please post it on the Eclipse OHF newsgroup.

The newsgroups is located at **news://news.eclipse.org/eclipse.technology.ohf**
You can request a password at: **http://www.eclipse.org/newsgroups/main.html**

5. Glossary

Apache Axis:

INSERT HERE

Eclipse:

Eclipse is an open source framework and platform for building software. The OHF Bridge uses the latest server-side OSGi technologies that will be included in the upcoming release of Eclipse 3.2. Specifically, OHF Bridge utilizes the server-side Equinox/OSGi technology, which is currently an incubator project within Eclipse. For more information, please go to http://www.eclipse.org/equinox/.

Electronic Medical Records (EMR):

EMRs are the patient medical records in electronic formats. The EMR data accepted by OHF Bridge fully conform to the HL7 international healthcare standard.

Interoperability Stack:

The Healthcare Repository is a physical datacenter that hosts patient medical documents. It has two separate components: the registry that hosts the metadata for patient and document search, and the repository that stores the contents of medical documents. In the event of patient search, the search is done in the registry; a list of matching patient is then returned. In the event of document retrieval, the registry accesses the repository, pulls the document and returns it. If the EMR application requests a creation, change or deletion of a document, the repository will first take the appropriate action and update the document, and then it will request a change in the metadata within the registry. A security component lies on top of both the repository and the registry and performs the permission handling.

Open Healthcare Framework (OHF):

OHF leverages the extensibility of Eclipse to create a set of standards and technologies to connect clinics and hospitals nationwide. For more information, please go to http://www.eclipse.org/ohf/.

OHF Bridge:

OHF Bridge, a component of the Eclipse Open Healthcare Framework, is the first step towards implementing the Open Healthcare Framework in non-Java applications. It is an open source, platform independent plug-in that can be implemented into existing EMR applications to add the capability of centralizing and searching of patient data. For more information, please go to http://wiki.eclipse.org/index.php/OHF#OHF_in_Action.

OSGi:

INSERT HERE

Tomcat:

INSERT HERE

Appendix A - Sample Code

The OHF Bridge provides a very simple API for patient search and document retrieval. This chapter describes the “hello world” code to search for a patient and to retrieve a document.

A.1 Search Patients

A.2 Retrieve Documents

A.3 Add Documents