i2b2 Cell Messaging

File Repository (FR) Cell

1. Table of Contents

i2b2 Cell Messaging 1

File Repository (FR) Cell 1

2. Document Version History 2

3. Introduction 3

3.1 The i2b2 Hive 3

3.2 i2b2 Messaging Overview 3

3.2.1 Message Header 4
3.2.2 Request Header 4
3.2.3 Response Header 4
3.2.4 Message Body 4

3.3 i2b2 XML Schema Definitions 5

4. The File Repository (FR) Cell Messaging Detail 6

4.1 Use Case 6

4.2 Messages 6

4.2.1 recvfile Request Message 6
4.2.2 recvfile Reponse Message 7
4.2.3 sendfile Request Message 7
4.2.4 Sendfile Response Message 8

5. Glossary 9

2. Document Version History

Date	Version	Description	Author(s)
8/11/2008	1.0	Initial Document	Mike Mendis

3. Introduction

This document gives an overview of i2b2 cell messaging as well as a more detailed description of message formats specific to the File Repository (FR) Cell.

3.1 The i2b2 Hive

Informatics for Integrating Biology and the Bedside (i2b2) is one of the sponsored initiatives of the NIH Roadmap National Centers for Biomedical Computing (http://www.bisti.nih.gov/ncbc/). One goal of i2b2 is to produce a comprehensive set of software tools to enable clinical investigators to collect and manage their project-related research data, including clinical and genomic data; that is, a software suite for the modern clinical research chart. Since different applications from different sources must be able to communicate with each other, a distributed computing model is needed, one that integrates multiple web-based applications in a standardized way.

The i2b2 hive and associated web services are the infrastructure used to create this integration. The hive is comprised of a collection of cells representing unique functional units. Cells in the hive have an array of roles, such as data storage, data analysis, ontology or identity management, natural language processing, and data conversion, derivation or de-identification. Each cell is a self-contained modular application that communicates with other cells via XML web services. A common i2b2 messaging protocol has been defined to enable the cells to interact with each other, sharing business logic, processes and data.

3.2 i2b2 Messaging Overview

All cells in the i2b2 hive communicate using standard, pre-defined i2b2 XML request and response messages.

A request message is sent from a client to a service and contains information, inside the top-level <request> tag, that allows the service to satisfy the request. The <request> tag contains a <message_header>, <request_header> and <message_body> as shown, below, in Figure 1.

The service sends back a response message, inside a top-level <response> tag, which informs the client about the status of the request and may also contain the actual results. The <response> tag contains its own <message_header>, <response_header> and <message_body> and it may optionally echo the request's <request_header> as shown, below, in Figure 1.

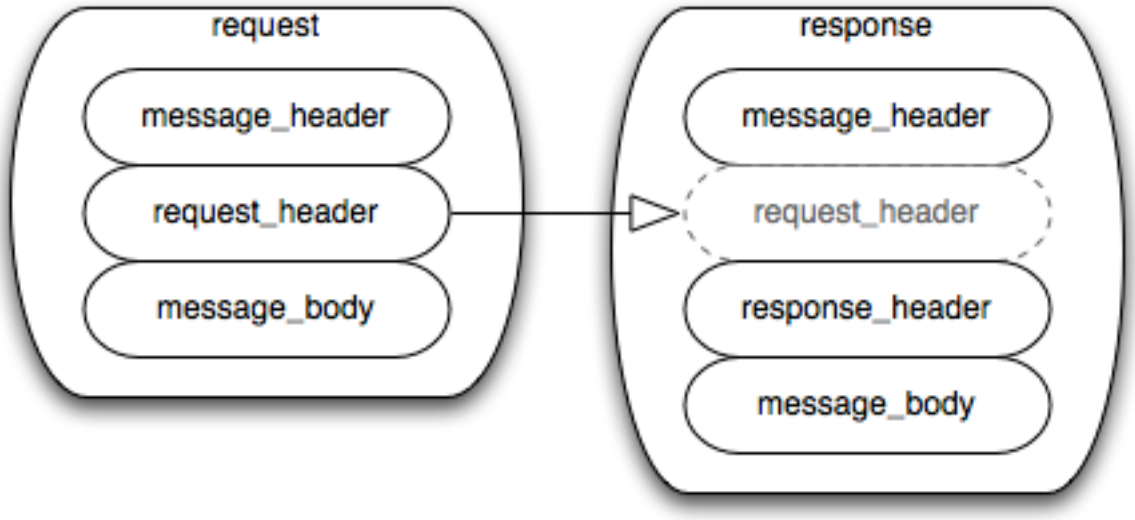

Figure 1: The basic structure of a request and response message. The request_header in the request can be echoed in the response.

3.2.1 Message Header

All requests are sent using a <request> tag and responses are returned using a <response> tag. The same <message_header> tag is used for both. Both request and response messages contain this <message_header> tag which has control information such as sending application, receiving application and message type.

3.2.2 Request Header

The request must contain a <request_header> tag which includes information about how to process a request such as the amount of time it is willing to wait for a response. The <request_header> tag may optionally be echoed back in the response.

3.2.3 Response Header

The response must include a <response_header> tag which includes general information about the response such as status and error messages or where to look for the results if they are not included with the response.

3.2.4 Message Body

Both request and response messages contain a <message_body> tag which may contain any well-formed xml. Individual cells may define cell-specific XML that will be put inside <message_body> tag. This cell-specific XML need not extend the i2b2 message schema since the i2b2 schema will allow insertion of tags from any namespace into the <message_body> tag.

3.3 i2b2 XML Schema Definitions

The i2b2 XML schema consists of three XSD files:

- **i2b2.xsd**
  This schema defines the type for the <message_header> and <message_body> tags described, in sections 0 and 0. This schema is included in the i2b2_request.xsd and the i2b2_response.xsd.

- **i2b2_request.xsd**
  This schema defines the type for the top-level <request> tag and the <request_header> tag described in section 0. It is used for validating i2b2 request messages.

- **i2b2_response.xsd**
  This schema defines the type for the top-level <response> tag and the <response_header> tag described in section 0. It is used for validating i2b2 response messages.

More details about the <request>, <response>, <message_header>, <request_header> and <response_header> tags can be found in a separate document describing the generic i2b2 message. The remainder of this document describes the contents of the <message_body> for the Project Management (PM) Cell.

4. The File Repository (FR) Cell Messaging Detail

The File Repository (FR) cell has two basic functions: to manage files and metadata associated with the file and to send and retrieve the files.

4.1 Use Case

4.2 Messages

4.2.1 recvfile Request Message

If the file does not need any special role access, than the client can go directly to the url to retrieve the file, otherwise the client can send recvfile request message and have the FR server send back the file.

```
<message_body>
        <recvfile_request>
                <filename>/oasis/ABT001B/brain_324.dcm</filename>
         </recvfile_request>
 </message_body>
```

A recvfile message is sent by a client application or another cell service to retrieve a file. The 'filename' attribute is used to specify the exact location of the file. The username, password and project_id from the i2b2 header are used for authentication.

The sequence of events is as follows:

1. Client requests a file (recvfile)

   These attributes come from the header and are used by the PM cell for authentication and role access:

   Username = a valid user logon
   Password = the password associated with the user
   Project_id = the name of the project location being accessed

2. The FR server performs the following steps:

a. Authentication: Verify that the user is valid within the domain and the project provided.

b. Authorization: Based on the role provided from the PM cell, the FR cell will verify that role has permission to access that file

d. If the role is allowed access to the file and the file exists, the file will be returned back to the requester via SOAP response attachment.

4.2.2 recvfile Reponse Message

```
  <message_body>
        <recvfile_response>
            <file size="3623" name="temp15151.xml" overwrite="true"
algorithm="MD5" hash="81100cfcb11417570d613125da90ec65" date="2008-08-
20T13:46:36.000-04:00" />
        </recvfile_response>
    </message_body>
```

A recvfile response message contains the status information associated with the file and also the file as an attachment

- date = The date the file was created/last modified
- size = The size of the file
- name = The name of the file
- algorithm = Algorithm to verify the file on (MD5, SHA, ect.)
- Hash = the hash based on the algorithm used
- overwrite = true/false if destination file should be overwritten

4.2.3 sendfile Request Message

```
    <message_body>
        <sendfile_request>
            <upload_file size="3623" name="temp15151.xml" overwrite="true"
algorithm="MD5" hash="81100cfcb11417570d613125da90ec65" date="2008-08-
20T13:46:36.000-04:00" />
        </sendfile_request>
    </message_body>
```

A sendfile message is sent by a client application to save a file and the associated data.

- date = The date the file was created/last modified
- size = The size of the file
- name = The name of the file
- algorithm = Algorithm to verify the file on (MD5, SHA, ect.)

- Hash = the hash based on the algorithm used
- overwrite = true/false if destination file should be overwritten

The sequence of events is as follows:

1. Client sends a file directly to the FR cell.

4.2.4 Sendfile Response Message

Response message for a valid user request:

In the FR response message, if the file was able to be saved

A response message for an valid file request:

```
<response_header>
      <result_status>
         <status type="DONE">file was successfully saved</status>
      </result_status>
</response_header>
```

In the cases were a error occurred due to the following possible events:

- Role provided is not valid for the user provided
- Username/password/project_id returned a error from the PM
- File_create_date is not valid
- File_modify_date is not valid
- File_size was not same as actual size
- Error occurred while saving file
- General server error
- File exists but "overwrite" set to false

A response message for an invalid file request:

```
<response_header>
      <result_status>
         <status type="ERROR">patient_id provided does not
exist!</status>
      </result_status>
</response_header>
```

5. Glossary