Informatics for Integrating Biology and the Bedside

i2b2

Software Documentation

i2b2 Cell Messaging

Identity Management (IM) Cell

Document Version:	*1.7.08-003*
i2b2 Software Version:	*1.7.08*

TABLE OF CONTENTS

DOCUMENT MANAGEMENT

Revision Number	Date	Author	Description of change
1.7.0	02/13/13	Mike Mendis	Created 1.7 version of document
1.7.00-001	08/11/2015	Janice Donahoe	Fixed spelling and grammar issues.
1.7.08-002	07/26/2016	S. Wayne Chan	Added 4 DBlookup Messages. Corrected location of the Validate_Site_ID Request & Response sections. Added examples for PDO_REQUEST Request & Response Messages. Added the Glossary chapter.
1.7.08-003	10/06/2016	Janice Donahoe	Updated documentation on the DB Lookup Messages.

1 INTRODUCTION

This document gives an overview of i2b2 cell messaging as well as a more detailed description of message formats specific to the **Identity Management (IM) Cell**.

1.1 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.bist.nigh.gov/ncbc/). One of the goals 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.

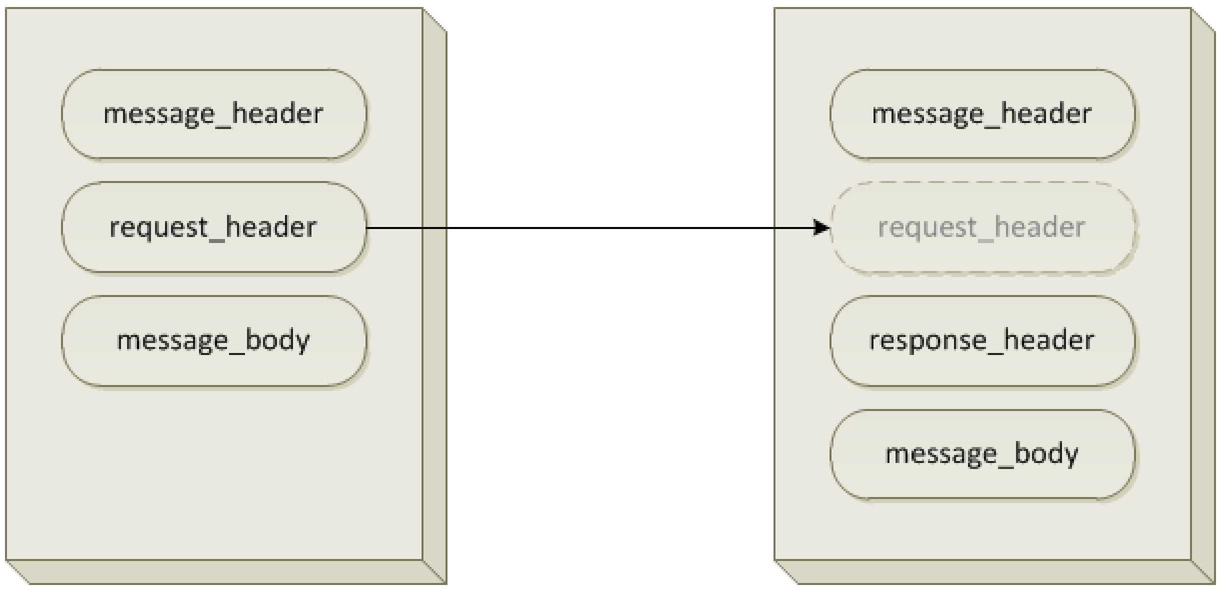
1.2 i2b2 Messaging Overview

All cells in the **i2b2 hive** must communicate using standard *i2b2 XML messages*. This message specifies certain properties that are common to all cells and are essential to the administration tasks associated with sending, receiving and processing 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*>.

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*>.

The following image illustrates the basic top-level elements contained within the request and response messages.



1.2.1 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 the request and response message contain this <*message_header*> tag which has control information such as the sending application, receiving application and the message type.

1.2.2 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.

1.2.3 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.

1.2.4 Message Body

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

1.3 i2b2 XML Schema Definitions

The i2b2 XML schema consists of three XSD files.

1.3.1 i2b2.xsd

This schema defines the type for the <*message_header*> and <*message_body*> tags. This schema is included in the *i2b2_request.xsd* and the *i2b2_response.xsd*.

1.3.2 i2b2_request.xsd

This schema defines the type for the top-level <*request*> tag and the <*request_header*> tag. It is used for validating i2b2 request messages.

1.3.3 i2b2_response.xsd

This schema defines the type for the top-level <*response*> tag and the <*response_header*> tag. It is used for validating i2b2 response messages.

Additional Resources

Additional 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 Identity Management (IM) Cell.

2 IM CELL MESSAGING DETAIL

The **Identity Management (IM) Cell** is a core i2b2 Hive cell.

2.1 Use Case

The diagram below depicts common use cases that a user may perform with the IM cell.

2.1.1 Operations

The IM service is designed as a collection of operations, or use cases.

Service	Description
set_key	Sets an AES key for a specific project that is used to decrypt the encrypted data either sent or received.
is_key_set	Verify that a key has been set for a specific project.
pdo_request	Receive a list of site ids that are associated with the input list and that are associated with the project.
validate_site_id	Verify that a list of site IDs are associated with a specific project.
get_audit	Return an audit trail for a specific based on a user, project or site ids.
get_all_dblookups	Returns a list of all the entries (rows) in the IM_DB_LOOKUP table.
set_dblookup	Adds or updates a specific entry (row) in the IM_DB_LOOKUP table.
get_dblookup	Returns all the data for a specific entry (row) in the IM_DB_LOOKUP table.
delete_dblookup	Deletes a specific entry (row) in the IM_DB_LOOKUP database table.

2.1.2 Database Lookup Services / Messages

The following four services and messages that will be invoked when querying the **IM_DB_LOOKUP** table in the *I2B2Hive* schema of your i2b2 database.

Service	XML Message
IMService/**getAllDblookups**	get_all_dblookups
IMService/**setDblookup**	set_dblookup
IMService/**getDblookup**	get_dblookup
IMService/**deleteDblookup**	delete_dblookup

The details for each of these messages is covered in the subsequent sections.

2.1.2.1 Restrictions

The role of **ADMIN** is required when running any of the *DBLookup messages*. The IM Cell with verify with the PM Cell that user is an ADMIN and if they are not then an error message will be returned in the response message.

2.2 Messages

2.2.1 set_key

A **set_key** message will set the key for a specific project.

User information is provided in the <*message_header*>; roles will be provided by the **Project Management (PM) cell**.

An ADMIN user can set the key for any project, while a MANAGER can set the key for only those projects they are defined as a manager.

Setting an empty key will unset the key.

The keys are kept in memory and will need to be set every time the IM cell is started.

2.2.1.1 SET_KEY Request Message

```
<message_body>
```

```
        <ns4:set_key>
            <project_id>Demo</project_id>
            <key>i2b2demodatakey1</key>
        </ns4:set_key>
    </message_body>
```

2.2.1.2 SET_KEY Response Message

Response message success:

```
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
```

Response message error:

```
    <response_header>
        <result_status>
            <status type="ERROR">Access Denied</status>
        </result_status>
    </response_header>
```

2.2.2 is_key_set

An **is_key_set** message returns if a key has been set for a specific project.

User information is provided in the <*message_header*>; roles will be provided by the **Project Management (PM) cell**.

An ADMIN can check the status of all projects, while a MANAGER can check the status for only those projects they are defined as a manager.

No information needs to be passed to the service.

2.2.2.1 IS_KEY_SET Request Message

```
<message_body>
    <im:is_key_set>
        <project_id>Demo</project_id>
    </im:is_key_set>
</message_body>
```

2.2.2.2 IS_KEY_SET Response Message

Response message success:

Response message error:

```
<response_header>
    <result_status>
        <status type="ERROR">Access Denied</status>
    </result_status>
</response_header>
```

2.2.3 pdo_request

A **pdo_request** message returns all site ids associated with a project.

Only a user with *DATA_PROT* can use this service.

The request is forwarded to the Data Repository Cell (CRC) and the Project information is provided in the <*message_header*>; roles will be provided by the **Project Management (PM) cell**.

2.2.3.1 Populating Patient IDs

The **pdo_request** message is used to get a list of patients associated with a project based on a list provided.

The sequence of events is as follows:

1. The client sends a PDO message with a list of site ids, either encrypted or decrypted or i2b2 patient numbers.
2. The IM server performs the following steps:
    a. The PDO request will take the existing <*pid_list*> in the <*input_list*> and create a new PDO request that will have a <*input_list*> and the <*filter_list*> will be blank, and the <*output_list*> will only have a <*pid_list*>.
    b. This new PDO will be forwarded to the Data Repository Cell (CRC).
    c. The response from the CRC, will contain a list of PIDs, the PIDs will be validated against the IM database to return only PIDs associated with the project.
    d. The filtered out PIDs will be saved in the audit table.
3. The client receives a list of unencrypted site ids.

2.2.3.2 PDO_REQUEST MESSAGE

A **pdo_request** message uses the same XML as the pdo_request for the Data Repository Cell.

Example: (note: pid below is fictitious)

```
<message_body>
    <ns3:pdoheader>
        <patient_set_limit></patient_set_limit>
        <estimated_time>180000</estimated_time>
        <request_type>getPDO_fromInputList</request_type>
    </ns3:pdoheader>
    <ns3:request xsi:type="ns3:GetPDOFromInputList_requestType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
        <input_list>
            <pid_list>
                <pid source="Hospital_1">2000001961</pid>
            </pid_list>
        </input_list>
        <output_option names="asattributes">
            <pid_set select="using_input_list" onlykeys="true">
        </output_option>
    </ns3:request>
<message_body>
```

2.2.3.3 PDO_REQUEST RESPONSE MESSAGE

A **pdo_request** message uses the same XML as the pdo_response for the Data Repository Cell.

Example: (note: all patient_id's and patient_map_id's below are fictitious)

```
<response_header>
    <result_status>
        <status type="DONE">DONE</status>
    </result_status>
</response_header>
<message_body>
        <ns3:response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="ns3:patient_data_responseType">
            <ns2:patient_data>
                <ns2:pid_set>
                    <pid>
                        <patient_id status="A" source="HIVE">2000001961</patient_id>
                        <patient_map_id source="Hospital_1">2005001961</patient_id>
                        <patient_map_id source="Hospital_2">3005001961</patient_id>
                    </pid>
                    <pid>
                        <patient_id status="A" source="HIVE">11489986</patient_id>
                        <patient_map_id source="Hospital_2">4005001961</patient_id>
                    </pid>
                </ns2:pid_set>
                <ns2:patient_set />
            </ns2:patient_data>
        </ns3:response>
</message_body>
```

2.2.4 Validate_Site_Id

The **validateSiteId** message is sent to verify if the list of site ids are associated with a project. Only a user with the role of DATA_PROT can use this service. The Project information is provided in the <*message_header*>; roles will be provided by the **Project Management (PM) cell**.

2.2.4.1 Validate Site ID

To validate a list of site ids, the sequence of events is as follows:

{See a description or similar in PDO for comment}

1. The client sends a PDO message with a list of site ids that are either encrypted or decrypted.
2. The IM server performs the following steps:
    a. Parses out a list of all the PIDs from the request and decrypts the encrypted ones.
    b. These PIDs will be validated against the IM database to return only those PIDs associated with the project.
    c. The filtered out PIDs will be saved in the audit table.
3. The client receives a list of unencrypted site ids.

When the PIDs are validated against the IM database (sequence 2b above) the following SQL is used; where the list of site ids is stored in a temporary table.

```
SELECT DISTINCT m1.lcl_id,
            m1.lcl_site
FROM           (SELECT  global_id ,
                    lcl_site  ,
                    lcl_id    ,
                    lcl_status,
                    Row_number() over ( PARTITION BY lcl_site, lcl_id ORDER BY update_date,
mapping_id) AS new_id
            FROM     im_mpi_mapping
```

```
                )
                m1                  ,
                im_mpi_demographics d ,
                im_project_patients pp,
                im_project_sites ps   ,
                im_temp_site ts
        WHERE         m1.new_id       = 1
        AND          d.global_id     = m1.global_id
        AND          d.global_status = 'A'
        AND          m1.lcl_status   = 'A'
        AND          pp.global_id    = d.global_id
        AND          ps.project_id   = pp.project_id
        AND          m1.lcl_site     = ts.lcl_site
        AND          m1.lcl_id       = ts.lcl_id
```

2.2.4.2 Validate Site ID Request Message

The message body of the **validateSiteId** message is basically the same as the **pdo_request** message, both use the same XML as the pdo_request for the Data Repository Cell (CRC).

Example: (note: pid below is fictitious)

```
<message_body>
    <ns3:pdoheader>
        <patient_set_limit></patient_set_limit>
        <estimated_time>180000</estimated_time>
        <request_type>getPDO_fromInputList</request_type>
    </ns3:pdoheader>
    <ns3:request xsi:type="ns3:GetPDOFromInputList_requestType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
        <input_list>
            <pid_list>
                <pid source="Hospital_1">2000001987</pid>
                <pid source="Hospital_2">02160313</pid>
            </pid_list>
        </input_list>
        <output_option names="asattributes">
            <pid_set select="using_input_list" onlykeys="true">
        </output_option>
    </ns3:request>
<message_body>
```

2.2.4.3 Validate Site ID Response Message

The message body of the **validateSiteId** response message is basically similar to the **pdo_request** response message, both use the same XML as the pdo_request for the Data Repository Cell (CRC).

Example: (note: all patient_id's and patient_map_id's below are fictitious)

```
<response_header>
    <result_status>
        <status type="DONE">IM processing completed</status>
    </result_status>
</response_header>
<message_body>
        <ns3:response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="ns3:patient_data_responseType">
            <ns2:patient_data>
                <ns2:pid_set>
                    <pid>
                        <patient_map_id source="Hospital_1">2000001987</patient_id>
                        <patient_map_id source="Hospital_2">02160313</patient_id>
                    </pid>
                </ns2:pid_set>
            </ns2:patient_data>
        </ns3:response>
</message_body>
```

2.2.5 get_audit

The **get_audit** message will get the audit.

The user information is provided in the <*message_header*>; roles will be provided by the **Project Management (PM) cell**.

An ADMIN user can get an audit for any project or user; while a MANAGER can retrieve the audit information they are a manager for.

Can filter by project, user and site information. The project comes from the <*message_header*> and is required.

The min and max is used for paging number of audits returned.

2.2.5.1 get_audit Request Message

```
<message_body>
    <ns4:get_audit min="1" max="100">
        <user_id>i2b2demo</user_id>
        <project_id>Demo</project_id>
        <source>EMP</source>
        <pid>123456</pid>
    </ns4:get_audit>
</message_body>
```

2.2.5.2 get_audit Response Message

Response message success:

```
<ns8:audits>
    <audit>
        <user_id>demo</user_id>
        <import_date>2013-02-27T09:29:21.000-05:00</import_date>
        <comment>test</comment>
        <source>BWH</source>
        <pid>2000001961</pid>
    </audit>
    <audit>
        <user_id>demo</user_id>
        <import_date>2013-02-27T09:29:21.000-05:00</import_date>
        <comment>test</comment>
        <source>BWH</source>
        <pid>11489986</pid>
    </audit>
</ns8:audits>
```

Response message error:

```
<response_header>
    <result_status>
```

```
            <status type="ERROR">Access Denied</status>
        </result_status>
    </response_header>
```

2.2.6 get_all_dblookups

As part of the **GetAllDBLookup** *service*, the client application will send a **get_all_dblookups** *message* to retrieve a list of all the database connections setup for the IM cell.

2.2.6.1 Processing by the IM Cell

The **get_all_dblookups** message is used to return a list of all the entries in the IM_DB_LOOKUP table. The sequence of events for this process are:

STEP 1: Send Request Message

The client sends a **request message** to the IM Cell asking for a list of all entries (rows) in the IM_DB_LOOKUP table.

STEP 2: Verify User Access

The IM Cell will send a request message to the PM Cell in order verify the user has the appropriate level of access.

- User is an Admin: the IM will continue processing the request.
- User is not an Admin: an error message will be returned.

STEP 3: Query the i2b2 Database

A **select query** is sent to the i2b2 database to retrieve all rows in the **IM_DB_LOOKUP** table that meet the following criteria.

1. The **C_DOMAIN_ID** in the IM_DB_LOOKUP table matches the **<domain> value** in the request message.

   Example:

C_DOMAIN_ID value *(IM_DB_LOOKUP Table)*		**<domain> value** *(request message)*
i2b2demo	=	i2b2demo

2. The **C_OWNER_ID** in the IM_DB_LOOKUP table either matches the **<username> value** in the request message or contains an "@".

STEP 3: Send Response Message

The IM sends a response message to the Client. The message contains a list of all the entries in the IM_DB_LOOKUP table that met the above criteria.

2.2.6.2 Message Structure

The **get_all_dblookups** *request / response* messages follow the standard i2b2 messaging structure. This section and the ones that follow contain additional information that is specific to the get_all_dblookups messages.

Note

For additional information please see the *Messaging Overview* section within this document.

The request and response message structure for the get_all_dblookups service is divided into three parts:

1. Invocation URL
2. Request Message
3. Response Message

For the request message, the <*request*> and *invocation URL* sections are required, and for the response message, the <*response*> and <*result_status*> sections are required.

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getAllDbloo
kups</redirect_url>
        </proxy>
        ...
    </message_header>
    <request_header>
        ...
    </request_header>
    <message_body>
        ...
    </message_body>
</i2b2:request>
```

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="value">Status Message</status>
        </result_status>
    </response_header>
    <message_body>
        ...
    </message_body>
</ns5:response>
```

2.2.6.2.1 Message Elements

Element Name	Description

request	The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base **RequestType *object***.
invocation url	The form of the message invocation URL is as follows: http://[*ipAddress*]:[*port#*]/i2b2/services/IMService/getAllDblookups
response	The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base **ResponseType *object*** containing a **StatusType *attribute***.
result_status	The <response_header> contains a <result_status> element, which will carry the status of the request. There are several types of statuses available: • DONE • ERROR • FATAL_ERROR • WARNING • INFO

2.2.6.3 Request Message: get_all_dblookups

```
<message_body>
    <ns4:get_all_dblookups type="default" />
</message_body>
```

2.2.6.3.1 Attributes

The only **attribute** available for *get_all_dblookups* request messages is:

Attribute Name	**Description**
type	Always set to "default".

2.2.6.3.2 XML Elements

<get_all_dblookups>	**Container for get_all_dblookups data request**

2.2.6.4 Response Message: get_all_dblookups

```
<response_header>
    <result_status>
        <status type="value">Status Message</status>
    </result_status>
</response_header>
<message_body>
    <ns3:dblookups>
        <dblookup project_path="value">
            <domain_id>value</domain_id>
            <owner_id>value</owner_id>
            <db_fullschema>value</db_fullschema>
            <db_datasource>value</db_datasource>
            <db_servertype>value</db_servertype>
            <db_nicename>value</db_nicename>
        </dblookup>
    </ns3:dblookups>
</message_body>
```

2.2.6.4.1 XML Elements

<dblookups>	**Container that wraps the <dblookup> container returned in the response message.** **The service will return all records in the IM_DB_LOOKUP table therefore this container may contain multiple <dblookup> containers.**

<dblookup>	**Container that wraps an object which holds the data for a single record (row) in the IM_DB_LOOKUP table.**
project_path	Contains the data stored in the **c_project_path** column in the IM_DB_LOOKUP table. The path of the project is stored in this column.
<domain_id>	Contains the data stored in the **c_domain_id** column in the IM_DB_LOOKUP table. The domain in which a project belongs to is stored in this column.
<owner_id>	Contains the data stored in the **c_owner_id** column in the IM_DB_LOOKUP table. The owner of the project is stored in this column. The value may be "@".
<db_fullschema>	Contains the data stored in the **c_db_fullschema** column in the IM_DB_LOOKUP table. The name of the IM database / schema is stored in this column.
<db_datasource>	Contains the data stored in the **c_datasource** column in the IM_DB_LOOKUP table. The data source for this project is stored in this column. The value represents the connection configuration for the IM Cell to communicate with the database.
<db_servertype>	Contains the data stored in the **c_servertype** column in the IM_DB_LOOKUP table. The type of database is stored in this column. The value may be any of the supported database management systems (Oracle, PostgreSQL, or SQL Server).
<db_nicename>	Contains the data stored in the **c_nicename** column in the IM_DB_LOOKUP table. A simple name that used to easily identify the database is stored in this column.

✓ **Tip**

Please refer to the *IM_Architecture* document for additional information about the IM_DB_LOOKUP table in the i2b2 Database.

2.2.6.5 Use Cases

2.2.6.5.1 ADMIN User Requests All DB Lookups

The get_all_dblookups service sends a request message to the IM cell and generates the output based on the user's level of access and the data requested.

In this use case, a user who has the role of ‘*ADMIN*’ has requested a list of all the entries in the IM_DB_LOOKUP table. A response message will be returned with the requested data.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getAllDblookups</redir
ect_url>
        </proxy>
        ...
    </message_header>
    <request_header>
        ...
    </request_header>
    <message_body>
        <pm:get_all_dblookup>
        </pm:get_all_dblookup>
    </message_body>
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
    <message_body>
        <ns3:dblookups>
            <dblookup project_path="/Demo_Oracle/">
                <domain_id>i2b2demo</domain_id>
                <owner_id>@</owner_id>
                <db_fullschema>i2b2demodata</db_fullschema>
                <db_datasource>java:/QueryToolDemoDS</db_datasource>
                <db_servertype>ORACLE</db_servertype>
                <db_nicename>Demo</db_nicename>
            </dblookup>
            <dblookup project_path="/Demo_SQL/">
                <domain_id>i2b2demo</domain_id>
                <owner_id>@</owner_id>
                <db_fullschema>i2b2demodata.dbo</db_fullschema>
                <db_datasource>java:/QueryToolDemoMartSQLDS</db_datasource>
                <db_servertype>SQLSERVER</db_servertype>
                <db_nicename>Demo Mart</db_nicename>
            </dblookup>
        </ns3:dblookups>
    </message_body>
</ns5:response>
```

2.2.6.5.2 Non-ADMIN User Requests All DB Lookups

The get_all_dblookups service sends a request message to the IM cell and generates the output based on the user’s level of access and the data requested.

In this use case, a user has requested a list of all the entries in the IM_DB_LOOKUP table, however they do not have the ‘ADMIN’ role. Therefore, the response message will be returned with an error message instead of the list of database connections.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getAllDblookups</redir
ect_url>
        </proxy>
        …
    </message_header>
    <request_header>
        …
    </request_header>
    <message_body>
        <pm:get_all_dblookup>
        </pm:get_all_dblookup>
    </message_body>
```

```
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="ERROR">Access denied, user not an admin!"</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.7 set_dblookup

As part of the **setDblookup** *service*, the client application will send a **set_dblookup** *message* to either add a new database connection or update an existing one.

2.2.7.1 Processing by the IM Cell

The **set_dblookup** message is used to either add a new entry or update an existing entry in the IM_DB_LOOKUP table. The sequence of events for this process are:

STEP 1: Verify User Access

The IM Cell will verify the user has the appropriate level of access.

- User is not an Admin: an error message will be returned.
- User is an Admin: the IM will continue processing the request.

STEP 2: Query the i2b2 Database

An **update query** is sent to the i2b2 database to either ***update an existing row*** in the IM_DB_LOOKUP table or ***add a new row***. The criteria to find an existing entry in the IM_DB_LOOKUP table.

1. The **C_DOMAIN_ID** in the IM_DB_LOOKUP table matches the **<domain> value** in the request message.

   Example:

C_DOMAIN_ID value *(IM_DB_LOOKUP Table)*		**<domain> value** *(request message)*
i2b2demo	**=**	i2b2demo

2. The **C_OWNER_ID** in the IM_DB_LOOKUP table either matches the **<username>** value in the request message or contains an "@".

3. The **C_PROJECT_PATH** in the IM_DB_LOOKUP table matches the **<project_path> value** from the request *message body attribute* in the request message.

   Example:

C_PROJECT_PATH value *(IM_DB_LOOKUP Table)*		*<set_dblookup* **project_path=""**> **value** *(request message)*
i2b2demo	**=**	Test20160518

Existing Entry

If a match is found, then the columns of that existing entry will be updated according to the corresponding parameter values in the request message.

New Entry

If a match is not found, then a new entry with the provided values will be added to the table.

STEP 3: Send Response Message

Once the update or add is completed, the IM sends a response message to the Client. The message simply lets the Client know the process has been completed.

2.2.7.2 Message Structure

The **set_dblookup** *request / response* messages follow the standard i2b2 messaging structure. This section and the ones that follow contain additional information that is specific to the set_dblookup messages.

> **Note**
>
> For additional information please see the *Messaging Overview* section within this document.

The request and response message structure for the set_dblookup service is divided into three parts:

1. Invocation URL
2. Request Message
3. Response Message

For the **request message**, the <*request*> and *invocation URL* sections are required, and for the **response message**, the <*response*> and <*result_status*> sections are required.

```
<i2b2:request>
    <message_header>
        <proxy>
    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/setDblookup</redirect_url>
        </proxy>
        …
    </message_header>
    <request_header>
        …
    </request_header>
    <message_body>
```

```
        ...
    </message_body>
</i2b2:request>


<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
    <message_body>
        ...
    </message_body>
</ns5:response>
```

2.2.7.2.1 Message Elements

Element Name	Description
request	The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base **RequestType *object***.
invocation url	The form of the message invocation URL is as follows: http://[*ipAddress*]:[*port#*]/i2b2/services/IMService/setDblookup
response	The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base **ResponseType *object*** containing a **StatusType *attribute***.
result_status	The <response_header> contains a <result_status> element, which will carry the status of the request. There are several types of statuses available: • DONE • ERROR • FATAL_ERROR • WARNING • INFO

2.2.7.3 Request Message: set_dblookup

```
<message_body>
    <ns4:set_dblookup project_path="/test20160518/” />
</message_body>
```

Note

The value of “/test20160518/” is simply an example intended to help illustrate this request message.

Example:

```
<message_body>
    <pm:set_dblookup project_path="test20160518/”>
            <domain_id>i2b2demo</ domain_id>
            <owner_id>@</owner_id>
            <db_fullschema>i2b2demodata</db_fullschema>
            <db_datasource>java:/QueryToolDemoDS</db_datasource>
            <db_servertype>ORACLE </db_servertype>
            <db_nicename>Demo</db_nicename>
            <db_tooltip>testing, …, 1, 2, 3, 4</db_tooltip>
            <comment>Just a test project</comment>
            <staus_cd></staus_cd>
    <pm:set_ dblookup>
</message_body>
```

2.2.7.3.1 Attributes

The only **attribute** available for *set_dblookup* request messages is:

Attribute Name	Description

project_path	The value entered here is used to identify an existing entry in the IM_DB_LOOKUP table.

2.2.7.3.2 XML Elements

<set_dblookup>	**Container that wraps an object which holds the data for a single record (row) that is being added or updated in the IM_DB_LOOKUP table.**
project_path	Contains the data stored in the **c_project_path** column in the IM_DB_LOOKUP table. The path of the project is stored in this column.
<domain_id>	Contains the data stored in the **c_domain_id** column in the IM_DB_LOOKUP table. The domain in which a project belongs to is stored in this column.
<owner_id>	Contains the data stored in the **c_owner_id** column in the IM_DB_LOOKUP table. The owner of the project is stored in this column. The value may be "@".
<db_fullschema>	Contains the data stored in the **c_db_fullschema** column in the IM_DB_LOOKUP table. The name of the IM database / schema is stored in this column.
<db_datasource>	Contains the data stored in the **c_datasource** column in the IM_DB_LOOKUP table. The data source for this project is stored in this column. The value represents the connection configuration for the IM Cell to communicate with the database.
<db_servertype>	Contains the data stored in the **c_servertype** column in the IM_DB_LOOKUP table. The type of database is stored in this column. The value may be any of the supported database management systems (Oracle, PostgreSQL, or SQL Server).
<db_nicename>	Contains the data stored in the **c_nicename** column in the IM_DB_LOOKUP table. A simple name that used to easily identify the database is stored in this column.
<db_tooltip>	Contains the data stored in the **db_tooltip** column in the IM_DB_LOOKUP table. A longer, sometimes hierarchical representation of the "nicename" is stored in this column.
<comment>	Contains the data stored in the **c_comment** column in the IM_DB_LOOKUP table.
<status_cd>	Contains the data stored in the **c_status_cd** column in the IM_DB_LOOKUP table.

✓ **Tip**

Please refer to the *IM_Architecture* document for additional information about the IM_DB_LOOKUP table in the i2b2 Database.

2.2.7.3.3 Required Attributes and Elements

In order to process the request, the following attributes and elements cannot be empty or missing from the xml request message. Missing information will result in an error when the message is processed by the IM.

- project_path (attribute)
- domain_id
- owner_id
- db_fullschema
- db_datasource
- db_servertype
- db_nicename

2.2.7.4 Response Message: set_dblookup

```
<response_header>
    <result_status>
        <status type="DONE">IM processing completed</status>
    </result_status>
</response_header>
```

2.2.7.5 Use Cases

2.2.7.5.1 ADMIN User Requests to Add a New or Edit an Existing Record

The **set_dblookup** service sends a request message to the IM cell and processes the request based on the user's level of access and the data requested.

In this use case, a user who has the role of '*ADMIN*' has requested to either add a new or edit an existing record in the IM_DB_LOOKUP table. Once the record has been added or updated a response message with the appropriate status will be returned.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/setDblookup</redirect_
url>
        </proxy>
        …
    </message_header>
    <message_body>
        <pm:set_dblookup project_path="/test20160518/">
            <domain_id>i2b2demo</domain_id>
            <owner_id>@</owner_id>
            <db_fullschema>i2b2demodata</db_fullschema>
            <db_datasource>java:/QueryToolDemoDS</db_datasource>
            <db_servertype>ORACLE</db_servertype>
            <db_nicename>Test</db_nicename>
            <db_tooltip></db_tooltip>
            <comment></comment>
            <status_cd></status_cd>
        </pm:set_dblookup>
    </message_body>
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
```

```
            ...
        </message_header>
        <response_header>
            <result_status>
                <status type="DONE">IM processing completed</status>
            </result_status>
        </response_header>
    </ns5:response>
```

2.2.7.5.2 Non-ADMIN User Requests to Add a New or Edit an Existing Record

The **set_dblookup** service sends a request message to the IM cell and processes the request based on the user's level of access and the data requested.

In this use case, a user has requested to either add a new or edit an existing record in the IM_DB_LOOKUP table, however they do not have the 'ADMIN' role. Therefore, the response message will be returned with an error message instead of adding / editing the record.

Request Message:

```
    <i2b2:request>
        <message_header>
            <proxy>

        <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/setDblookup</redirect_
    url>
            </proxy>
            ...
        </message_header>
        <message_body>
            <pm:set_dblookup project_path="/test20160518/">
                <domain_id>i2b2demo</domain_id>
                <owner_id>@</owner_id>
                <db_fullschema>i2b2demodata</db_fullschema>
                <db_datasource>java:/QueryToolDemoDS</db_datasource>
                <db_servertype>ORACLE</db_servertype>
                <db_nicename>Test</db_nicename>
                <db_tooltip></db_tooltip>
                <comment></comment>
                <status_cd></status_cd>
            </pm:set_dblookup>
        </message_body>
    </i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.7.5.3 Required Information Not Sent in Request

The **set_dblookup** service sends a request message to the IM cell and processes the request based on the user's level of access and the data requested.

In this use case, a user with the appropriate level of access has requested to either add a new or edit an existing record in the IM_DB_LOOKUP table, however an attribute or key element is missing or empty. Therefore, the response message will be returned with an error message instead of adding / editing the record in the table.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/setDblookup</redirect_
url>
        </proxy>
        ...
    </message_header>
```

```
        <message_body>
            <pm:set_dblookup project_path="">
                <domain_id>i2b2demo</domain_id>
                <owner_id>@</owner_id>
                <db_fullschema>i2b2demodata</db_fullschema>
                <db_datasource>java:/QueryToolDemoDS</db_datasource>
                <db_servertype>ORACLE</db_servertype>
                <db_nicename>Test</db_nicename>
                <db_tooltip></db_tooltip>
                <comment></comment>
                <status_cd></status_cd>
            </pm:set_dblookup>
        </message_body>
    </i2b2:request>
```

In the example above, the value for the required *project_path* attribute is missing from the request message.

Response Message:

```
    <ns5:response>
        <message_header>
            ...
        </message_header>
        <response_header>
            <result_status>
                <status type="ERROR">'project_path', 'domain_id', 'owner_id', 'db_fullschema',
    'db_datasource', 'db_servertype', or 'db_nicename' can't be missing or blank!</status>
            </result_status>
        </response_header>
    </ns5:response>
```

2.2.8 get_dblookup

As part of the **getDblookup** *service*, the client application will send a **get_dblookup** *message* to either add a new database connection or update an existing one.

2.2.8.1 Processing by the IM Cell

The **get_dblookup** message is used to return data an existing entry in the IM_DB_LOOKUP table. The sequence of events for this process are:

STEP 1: Verify User Access

The IM Cell will verify the user has the appropriate level of access.

- User is not an Admin: an error message will be returned.
- User is an Admin: the IM will continue processing the request.

STEP 2: Query the i2b2 Database

A **select query** is sent to the i2b2 database to retrieve the data on a specific entry in the IM_DB_LOOKUP table. The criteria to find an existing entry in the IM_DB_LOOKUP table.

1. The **C_DOMAIN_ID** in the IM_DB_LOOKUP table matches the **<domain> value** in the request message.

   Example:

C_DOMAIN_ID value *(IM_DB_LOOKUP Table)*		**<domain> value** *(request message)*
i2b2demo	**=**	i2b2demo

2. The **C_OWNER_ID** in the IM_DB_LOOKUP table either matches the **<username>** value in the request message or contains an "@".

3. The **C_PROJECT_PATH** in the IM_DB_LOOKUP table matches the **project_path value** from the request *message body attribute* in the request message.

   Example:

C_PROJECT_PATH value *(IM_DB_LOOKUP Table)*		<*get_dblookup* **field="project_path" value=""**> *(request message)*
i2b2demo	=	test20160518

STEP 3: Send Response Message

The IM sends a response message to the Client. The message contains a list of all the entries in the IM_DB_LOOKUP table.

2.2.8.2 Message Structure

The **get_dblookup** *request / response* messages follow the standard i2b2 messaging structure. This section and the ones that follow contain additional information that is specific to the get_dblookup messages.

Note

For additional information please see the *Messaging Overview* section within this document.

The request and response message structure for the set_dblookup service is divided into three parts:

1. Invocation URL
2. Request Message
3. Response Message

For the request message, the <*request*> and *invocation URL* sections are required, and for the response message, the <*response*> and <*result_status*> sections are required.

```
<i2b2:request>
    <message_header>
        <proxy>
    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getDblooku
p</redirect_url>
        </proxy>
        ...
    </message_header>
    <request_header>
        ...
    </request_header>
    <message_body>
        ...
    </message_body>
</i2b2:request>


<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
    <message_body>
        ...
    </message_body>
</ns5:response>
```

2.2.8.2.1 Message Elements

Element Name	**Description**
request	The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base **RequestType *object***.
invocation url	The form of the message invocation URL is as follows: http://[*ipAddress*]:[*port#*]/i2b2/services/IMService/getDblookup
response	The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base **ResponseType *object*** containing a **StatusType *attribute***.
result_status	The <response_header> contains a <result_status> element, which will carry the status of the request. There are several types of statuses available:

	• DONE • ERROR • FATAL_ERROR • WARNING • INFO

2.2.8.3 Request Message: get_dblookup

```
<message_body>
    <pm:get_dblookup field="value" value="value" />
</message_body>
```

Note

The value of "/test20160518/" is simply an example intended to help illustrate this request message.

Example:

```
<message_body>
    <pm:get_dblookup field="project_path" value="/test20160518/" />
</message_body>
```

If the *'field'* attribute is omitted, then *"project_path"* will be used as the default.

Example:

```
<message_body>
    <pm:get_dblookup value="/test20160518/"/>
</message_body>
```

⚠ **Important Requirement**

An error will occur if the ‘field’ attribute is left blank or the ‘value’ attribute is missing or blank.

Examples: The following examples will result in an error when received by the IM Cell.

2.2.8.3.1 Attributes

The **attributes** available for *get_dblookup* request messages are:

Attribute Name	Description
field	The value for the “field” attribute is equivalent to the column name in the IM_DB_LOOKUP table. The only difference is the field does not have the leading ‘c_’ in its name. ***Example:*** field=“*project_path*” maps to the *c_project_path* column in the IM_DB_LOOKUP table.
value	The specific data in the specified field to look for when querying the IM_DB_LOOKUP table.

2.2.8.3.2 XML Elements

<get_dblookup>	**Container for get_dblookup data request**

✓ **Tip**

Please refer to the *IM_Architecture* document for additional information about the IM_DB_LOOKUP table in the i2b2 Database.

2.2.8.3.3 Required Attributes

In order to process the request, the following attributes cannot be empty or missing from the xml request message. Missing information will result in an error when the message is processed by the IM.

- field: attribute is missing its value (left blank)
- attribute: attribute missing or empty (left blank)

Examples

The following examples will result in an error when received by the IM Cell:

```
<ns4:get_dblookup field="project_path" />
<ns4:get_dblookup field="project_path" value="" />
<ns4:get_dblookup value="" />
<ns4:get_dblookup />
```

2.2.8.4 Response Message: get_dblookup

```
<response_header>
    <result_status>
        <status type="DONE">IM processing completed</status>
    </result_status>
</response_header>
<message_body>
    <ns4:dblookups>
        <dblookup project_path="test20160518/">
            <domain_id>i2b2demo</domain_id>
            <owner_id>@</owner_id>
            <db_fullschema>i2b2demodata</db_fullschema>
```

```
                <db_datasource>java:/QueryToolDemoDS</db_datasource>
                <db_servertype>ORACLE</db_servertype>
                <db_nicename>Demo</db_nicename>
                <db_tooltip>Demo Database Connection</db_tooltip>
                <comment>Demo Database used to demonstrate the software</comment>
                <entry_date>2016-10-05 13:00:00</entry_date>
                <change_date>2016-10-05 13:59:43</change_date>
            </dblookup>
        </ns4:dblookups>
    </message_body>
```

2.2.8.4.1 XML Elements

<dblookups>	**Container that wraps the <dblookup> container returned in the response message.** **The service will return all records in the IM_DB_LOOKUP table therefore this container may contain multiple <dblookup> containers.**

<dblookup>	**Container that wraps an object which holds the data for a single record (row) in the IM_DB_LOOKUP table.**
project_path	Contains the data stored in the **c_project_path** column in the IM_DB_LOOKUP table. The path of the project is stored in this column.
<domain_id>	Contains the data stored in the **c_domain_id** column in the IM_DB_LOOKUP table. The domain in which a project belongs to is stored in this column.
<owner_id>	Contains the data stored in the **c_owner_id** column in the IM_DB_LOOKUP table. The owner of the project is stored in this column. The value may be "@".
<db_fullschema>	Contains the data stored in the **c_db_fullschema** column in the IM_DB_LOOKUP table. The name of the IM database / schema is stored in this column.
<db_datasource>	Contains the data stored in the **c_datasource** column in the IM_DB_LOOKUP table. The data source for this project is stored in this column. The value represents the connection configuration for the IM Cell to communicate with the database.

<db_servertype>	Contains the data stored in the **c_servertype** column in the IM_DB_LOOKUP table. The type of database is stored in this column. The value may be any of the supported database management systems (Oracle, PostgreSQL, or SQL Server).
<db_nicename>	Contains the data stored in the **c_nicename** column in the IM_DB_LOOKUP table. A simple name that used to easily identify the database is stored in this column.

✓ **Tip**

Please refer to the *IM_Architecture* document for additional information about the IM_DB_LOOKUP table in the i2b2 Database.

2.2.8.5 Use Cases

2.2.8.5.1 ADMIN User Requests Data on a Specific Database Connection

The **get_dblookup** service sends a request message to the IM cell and generates the output based on the user's level of access and the data requested.

In this use case, a user who has the role of '*ADMIN*' has requested data on a specific entry in the IM_DB_LOOKUP table. The response message will be returned with the requested data.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getDblookup</redirect_
url>
        </proxy>
        ...
    </message_header>
    <request_header>
```

```
            ...
        </request_header>
        <message_body>
            <pm: get_dblookup value="/test20160518/"/>
        </message_body>
    </i2b2:request>
```

Response Message:

```
    <ns5:response>
        <message_header>
            ...
        </message_header>
        <response_header>
            <result_status>
                <status type="DONE">IM processing completed</status>
            </result_status>
        </response_header>
        <message_body>
            <ns3:dblookups>
                <dblookup project_path="/test20160518/">
                    <domain_id>i2b2demo</domain_id>
                    <owner_id>@</owner_id>
                    <db_fullschema>i2b2demodata</db_fullschema>
                    <db_datasource>java:/QueryToolDemoDS</db_datasource>
                    <db_servertype>ORACLE</db_servertype>
                    <db_nicename>Demo</db_nicename>
                </dblookup>
            </ns3:dblookups>
        </message_body>
    </ns5:response>
```

2.2.8.5.2 Non-ADMIN User Requests Data on a Specific Database Connection

The **get_dblookup** service sends a request message to the IM cell and generates the output based on the user's level of access and the data requested.

In this use case, a user has requested data on a specific entry in the IM_DB_LOOKUP table, however they do not have the 'ADMIN' role. Therefore, the response message will be returned with an error message instead of the database connection information.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getDblookup</redirect_
url>
        </proxy>
        ...
    </message_header>
    <request_header>
        ...
    </request_header>
    <message_body>
        <pm: get_dblookup value="/test20160518/"/>
    </message_body>
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="ERROR">Access denied, user not an admin!"</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.8.5.3 Requested Data Not Found

The **get_dblookup** service sends a request message to the IM cell and generates the output based on the user's level of access and the data requested.

In this use case, a user with the appropriate level of access has requested data on a specific entry in the IM_DB_LOOKUP table, however the requested data does not exist in the table. Therefore, the response message will be returned with an error message instead of the database connection information.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getDblookup</redirect_
url>
        </proxy>
        ...
    </message_header>
    <request_header>
        ...
    </request_header>
    <message_body>
        <pm: get_dblookup value="/demo-456/"/>
    </message_body>
</i2b2:request>
```

Response Message:

```
<response_header>
    <result_status>
        <status type="DONE">No dblookup row was found! - IM processing
completed</status>
    </result_status>
</response_header>
```

2.2.8.5.4 Required Information Not Sent in Request

The **get_dblookup** service sends a request message to the IM cell and generates the output based on the user's level of access and the data requested.

In this use case, a user with the appropriate level of access has requested data on a specific entry in the IM_DB_LOOKUP table, however the field attribute is empty or the value attribute is missing or empty. Therefore, the response message will be returned with an error message instead of the database connection information.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/getDblookup</redirect_
url>
        </proxy>
        …
    </message_header>
    <request_header>
        …
    </request_header>
    <message_body>
        <pm: get_dblookup field="project_path"/>
    </message_body>
</i2b2:request>
```

In the example above, the required *value* attribute is missing from the request message.

Response Message:

```
<response_header>
    <result_status>
        <status type="ERROR">'field' can't be blank, or 'value' can't be missing or
blank!</status>
    </result_status>
</response_header>
```

2.2.9 delete_dblookup

As part of the **deleteDblookup** *service*, the client application will send a **delete_dblookup** *message* to remove a specific record from the IM_DB_LOOKUP table.

2.2.9.1 Processing by the IM Cell

The **delete_dblookup** message is used to remove an existing entry from the IM_DB_LOOKUP table. The sequence of events for this process are:

STEP 1: Verify User Access

The IM Cell will verify the user has the appropriate level of access.

- User is not an Admin: an error message will be returned.
- User is an Admin: the IM will continue processing the request.

STEP 2: Query the i2b2 Database

A **delete query** is sent to the i2b2 database to ***delete a specific row*** in the IM_DB_LOOKUP table. The criteria to find an existing entry in the IM_DB_LOOKUP table.

1. The **C_DOMAIN_ID** in the IM_DB_LOOKUP table matches the **value** for the **domain_id *attribute*** in the request message.

   Example:

C_DOMAIN_ID value *(IM_DB_LOOKUP Table)*		*<delete_dblookup* **domain_id=""> value** *(request message)*
i2b2demo	**=**	i2b2demo

2. The **C_OWNER_ID** in the IM_DB_LOOKUP table either matches the **value** for the **user_id *attribute*** in the request message or contains an “@”.

C_OWNER_ID value *(IM_DB_LOOKUP Table)*		*<delete_dblookup* **user_id=""**> **value** *(request message)*
i2b2demo	=	@

3. The **C_PROJECT_PATH** in the IM_DB_LOOKUP table matches the **value** for the **project_path *attribute*** in the request message.

   Example:

C_PROJECT_PATH value *(IM_DB_LOOKUP Table)*		*<delete_dblookup* **project_path=" "**> **value** *(request message)*
i2b2demo	**=**	test20160518

STEP 3: Send Response Message

Once the deletion is completed, the IM sends a response message to the Client. The message simply lets the Client know the process has been completed.

2.2.9.2 Message Structure

The **delete_dblookup** *request / response* messages follow the standard i2b2 messaging structure. This section and the ones that follow contain additional information that is specific to the set_dblookup messages.

Note

For additional information please see the *Messaging Overview* section within this document.

The request and response message structure for the delete_dblookup service is divided into three parts:

1. Invocation URL
2. Request Message
3. Response Message

For the request message, the <*request*> and *invocation URL* sections are required, and for the response message, the <*response*> and <*result_status*> sections are required.

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/deleteDblo
okup</redirect_url>
        </proxy>
        …
    </message_header>
    <request_header>
        …
    </request_header>
    <message_body>
        …
    </message_body>
</i2b2:request>


<ns5:response>
    <message_header>
        …
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
    <message_body>
```

```
        ...
    </message_body>
</ns5:response>
```

2.2.9.2.1 Message Elements

Element Name	Description
request	The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base **RequestType *object***.
invocation url	The form of the message invocation URL is as follows: http://[*ipAddress*]:[*port#*]/i2b2/services/IMService/deleteDblookup
response	The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base **ResponseType *object*** containing a **StatusType *attribute***.
result_status	The <response_header> contains a <result_status> element, which will carry the status of the request. There are several types of statuses available: • DONE • ERROR • FATAL_ERROR • WARNING • INFO

2.2.9.3 Request Message: delete_dblookup

```
<message_body>
    <pm:delete_dblookup project_path="xyz" domain_id="xyz" owner_id="@"/>
</message_body>
```

2.2.9.3.1 Attributes

The **attributes** available for *delete_dblookup* request messages are:

Attribute Name	Description
project_path	Equivalent to the *c_project_path* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_project_path column for matching record(s).
domain_id	Equivalent to the *c_domain_id* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_domain_id column for matching record(s).
owner_id	Equivalent to the *c_owner_id* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_owner_id column for matching record(s).

Example:

The following example illustrates a typical message body for this request, with the attributes being *project_path*, *domain_id*, and *owner_id*:

```
<message_body>
    <pm:delete_dblookup project_path="/test20160518/" domain_id="i2b2demo"
owner_id="@"/>
</message_body>
```

2.2.9.3.2 XML Elements

<delete_dblookup>	Container for delete_dblookup data request

Tip

Please refer to the *IM_Architecture* document for additional information about the IM_DB_LOOKUP table in the i2b2 Database.

2.2.9.3.3 Required Attributes

In order to process the request, the following attributes cannot be empty or missing from the xml request message. Missing information will result in an error when the message is processed by the IM.

- project_path
- domain_id
- owner_id

2.2.9.4 Response Message delete_dblookup

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="value">message</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.9.5 Use Cases

2.2.9.5.1 ADMIN User Requests Deletion of Record

The **delete_dblookup** service sends a request message to the IM cell and processes the request based on the user's level of access and the data requested.

In this use case, a user who has the role of '*ADMIN*' has requested a specific record be deleted from the IM_DB_LOOKUP table. Once the record is deleted a response message with the appropriate status will be returned.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/deleteDblookup</redire
ct_url>
        </proxy>
        ...
    </message_header>
    <message_body>
        <pm:delete_dblookup project_path="/test20160518/" domain_id="i2b2demo"
owner_id="@"/>
    </message_body>
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="DONE">IM processing completed</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.9.5.2 Non-ADMIN User Requests Deletion of Record

The **delete_dblookup** service sends a request message to the IM cell and process the request based on the user's level of access and the data requested.

In this use case, a user has requested a specific record be deleted from the IM_DB_LOOKUP table, however they do not have the 'ADMIN' role. Therefore, the record will not be deleted and the response message will be returned with an error message.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>
            
    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/deleteDblookup</redire
ct_url>
        </proxy>
        ...
    </message_header>
    <message_body>
        <pm:delete_dblookup project_path="/test20160518/" domain_id="i2b2demo"
owner_id="@"/>
    </message_body>
</i2b2:request>
```

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="ERROR">Access denied, user not an admin!"</status>
        </result_status>
    </response_header>
</ns5:response>
```

2.2.9.5.3 Requested Record Doesn’t Exist

The **delete_dblookup** service sends a request message to the IM cell and process the request based on the user’s level of access and the data requested.

In this use case, a user with the appropriate level of access has requested a specific record be deleted from the IM_DB_LOOKUP table, however the record does not exist in the table. Therefore, the response message will be returned with an error message.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/deleteDblookup</redire
ct_url>
        </proxy>
        …
    </message_header>
    <message_body>
        <pm:delete_dblookup project_path="/test20160518/" domain_id="i2b2demo"
owner_id="@"/>
    </message_body>
</i2b2:request>
```

Response Message:

```
<response_header>
    <result_status>
        <status type="DONE”>no dblookup row was deleted (could be due to no target row
found)! - IM processing completed</status>
    </result_status>
</response_header>
```

2.2.9.5.4 Required Information Not Sent in Request

The **delete_dblookup** service sends a request message to the IM cell and process the request based on the user’s level of access and the data requested.

In this use case, a user with the appropriate level of access has requested a specific record be deleted from the IM_DB_LOOKUP table, however a required parameter or value is missing. Therefore, the response message will be returned with an error message.

Request Message:

```
<i2b2:request>
    <message_header>
        <proxy>

    <redirect_url>http://[ipAddress]:[port#]/i2b2/services/IMService/deleteDblookup</redire
ct_url>
        </proxy>
        ...
    </message_header>
    <message_body>
        <pm:delete_dblookup domain_id="i2b2demo" owner_id="@"/>
    </message_body>
</i2b2:request>
```

In the example above, the required *project_path* attribute is missing from the request message.

Response Message:

```
<ns5:response>
    <message_header>
        ...
    </message_header>
    <response_header>
        <result_status>
            <status type="ERROR">
                'project_path', or 'domain_id', 'owner_id' can't be missing or blank!"
            </status>
        </result_status>
    </response_header>
</ns5:response>
```

3 IM CELL XML SCHEMA DEFINITIONS

The **Identity Management XML schema** consists of the following XSD files that define the <*message_body*> for the entire IM cell.

3.1 IM.xsd

This schema is used to describe the components that are common to the request and response messages.

3.2 IM_QRY.xsd

Describes the response message body for all the operations described in section 3 of this document.

3.3 IM_RESP.xsd

Describes the response message body for all the operations described in section 3 of this document.

4 GLOSSARY

4.1 Message Tags & Attribute Definitions

4.1.1 IM_QRY.xsd

<request>	**Container for request information**

<get_all_dblookups>	**Container for get_all_dblookups data request**
type	Always set to "default".

<set_dblookup>	**Container that wraps an object which holds the data for a single record (row) that is being added or updated in the IM_DB_LOOKUP table.**
project_path	Contains the data stored in the **c_project_path** column in the IM_DB_LOOKUP table. The path of the project is stored in this column.
<domain_id>	Contains the data stored in the **c_domain_id** column in the IM_DB_LOOKUP table. The domain in which a project belongs to is stored in this column.
<owner_id>	Contains the data stored in the **c_owner_id** column in the IM_DB_LOOKUP table. The owner of the project is stored in this column. The value may be "@".
<db_fullschema>	Contains the data stored in the **c_db_fullschema** column in the IM_DB_LOOKUP table. The name of the IM database / schema is stored in this column.
<db_datasource>	Contains the data stored in the **c_datasource** column in the IM_DB_LOOKUP table. The data source for this project is stored in this column. The value represents the connection configuration for the IM Cell to communicate with the database.
<db_servertype>	Contains the data stored in the **c_servertype** column in the IM_DB_LOOKUP table. The type of database is stored in this column. The value may be any of the supported database management systems (Oracle, PostgreSQL, or SQL Server).
<db_nicename>	Contains the data stored in the **c_nicename** column in the IM_DB_LOOKUP table. A simple name that used to easily identify the database is stored in this column.

<db_tooltip>	Contains the data stored in the **db_tooltip** column in the IM_DB_LOOKUP table. A longer, sometimes hierarchical representation of the "nicename" is stored in this column.
<comment>	Contains the data stored in the **c_comment** column in the IM_DB_LOOKUP table.
<status_cd>	Contains the data stored in the **c_status_cd** column in the IM_DB_LOOKUP table.

<get_dblookup>	**Container for get_dblookup data request**
field	The "field" is equivalent to the column name in the IM_DB_LOOKUP table. The only difference is the field does not have the leading 'c_' in its name. ***Example:*** field="*project_path*" maps to the *c_project_path* column in the IM_DB_LOOKUP table.
value	The specific data in the specified field to look for when querying the IM_DB_LOOKUP table.

<delete_dblookup>	**Container for delete_dblookup data request**
project_path	Equivalent to the *c_project_path* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_project_path column for matching record(s).
domain_id	Equivalent to the *c_domain_id* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_domain_id column for matching record(s).
owner_id	Equivalent to the *c_owner_id* column in the IM_DB_LOOKUP table. The value sent for this parameter will be used to search the c_owner_id column for matching record(s).

4.1.2 IM_RESP.xsd

<response>	**Container for response information**

<dblookups>	**Container that wraps the <dblookup> container returned in the response message.** **The service will return all records in the IM_DB_LOOKUP table therefore this container may contain multiple <dblookup> containers.**

<dblookup>	**Container that wraps an object which holds the data for a single record (row) in the IM_DB_LOOKUP table.**
project_path	Contains the data stored in the **c_project_path** column in the IM_DB_LOOKUP table. The path of the project is stored in this column.
<domain_id>	Contains the data stored in the **c_domain_id** column in the IM_DB_LOOKUP table. The domain in which a project belongs to is stored in this column.
<owner_id>	Contains the data stored in the **c_owner_id** column in the IM_DB_LOOKUP table. The owner of the project is stored in this column. The value may be "@".
<db_fullschema>	Contains the data stored in the **c_db_fullschema** column in the IM_DB_LOOKUP table. The name of the IM database / schema is stored in this column.
<db_datasource>	Contains the data stored in the **c_datasource** column in the IM_DB_LOOKUP table. The data source for this project is stored in this column. The value represents the connection configuration for the IM Cell to communicate with the database.
<db_servertype>	Contains the data stored in the **c_servertype** column in the IM_DB_LOOKUP table. The type of database is stored in this column. The value may be any of the supported database management systems (Oracle, PostgreSQL, or SQL Server).
<db_nicename>	Contains the data stored in the **c_nicename** column in the IM_DB_LOOKUP table. A simple name that used to easily identify the database is stored in this column.