i2b2 Cell Messaging

Ontology Management (ONT) Cell (1.1)

1. Table of Contents

2. Document Version History

Date	Version	Description	Author(s)
12/08/2006	0.9	Pre-Release Version	Kristel Hackett
6/11/2007	1.0	Version 1.0	Vivian Gainer
07/30/2007	1.01	Minor updates	Lori Phillips
11/01/2007	1.1	Version 1.1	Lori Phillips

3. Introduction

This document gives an overview of i2b2 cell messaging as well as a more detailed description of message formats specific to the Ontology (ONT) 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 it's own <message_header>, <response_header> and <message_body> and it may optionally echo the request's <request_header> as shown, below, in Figure 1.

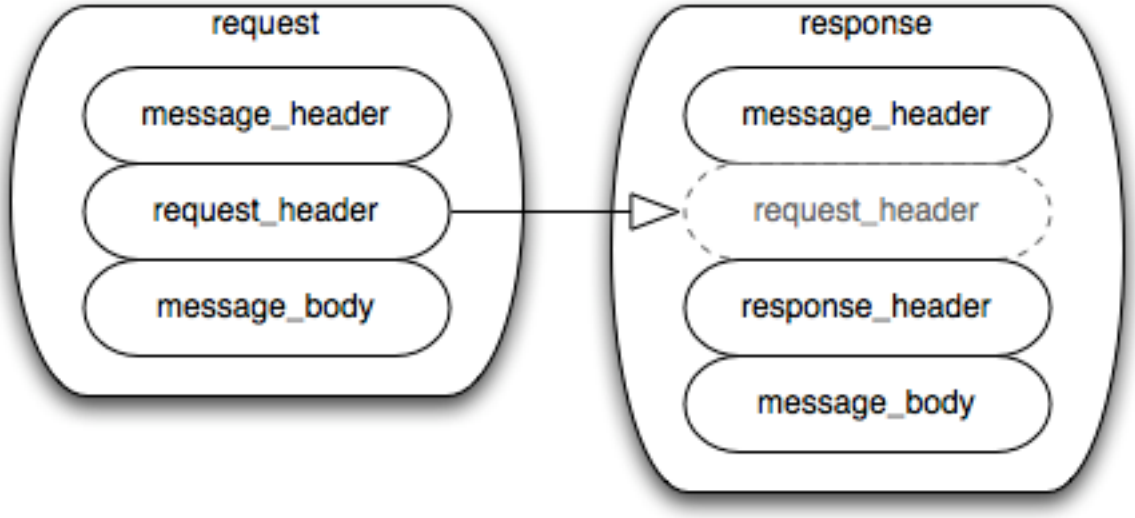


Figure 1: The basic structure of a request and response message. The request_header in the request can be echoed in the response.

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. 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. 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. 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 Ontology Management (ONT) Cell.

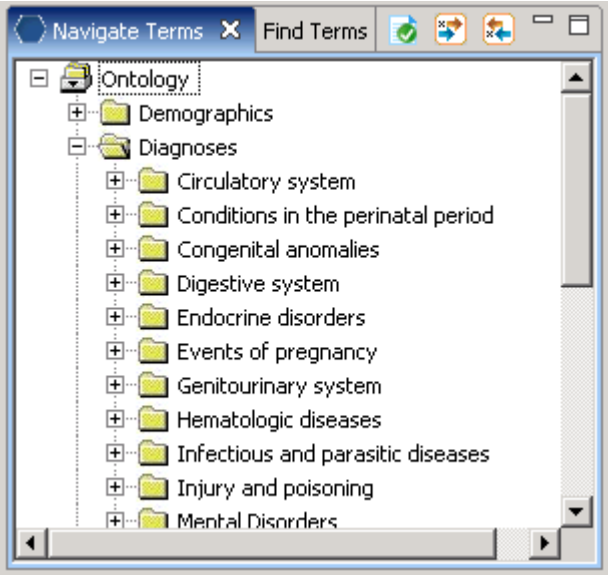
4. The Ontology Management (ONT) Cell Messaging Detail

Definitions for the i2b2 vocabularies reside in the Ontology Management (ONT) Cell. This cell contains concepts and information about relationships between concepts for the entire hive. It is accessed by other cells to give semantic meaning to data.

Vocabularies in the ONT cell are organized in hierarchical structures that represent the relationships between terms. The top levels in the hierarchy are called the 'parents' or 'roots', with the lower levels being their 'children' . Elements occurring on the same level are known as 'siblings'. A level in a hierarchy is sometimes referred to as a 'node', and a group of related data is called a 'category'.

A category is defined as a set of data for which there is a common rule or rules for querying against the Clinical Research Chart (CRC) . A category is usually represented visually as a table of terms. An example of a category is the Diagnosis category shown in the diagram below, which consists of a table of diagnoses terms and uses a single rule to build all diagnosis queries.

.



Vocabularies in the ONT cell may originate from different sources, and the codes from each source are distinguished from the others by a unique prefix which is appended to the source code. Each distinct vocabulary and their associated codes is called a scheme.

The ONT service is designed as a collection of operations, or use cases:

get_categories: returns a list of tables or categories available for a given user. These are the parent or root nodes. The top level consists of all the tables (categories) a particular user has permission to see.

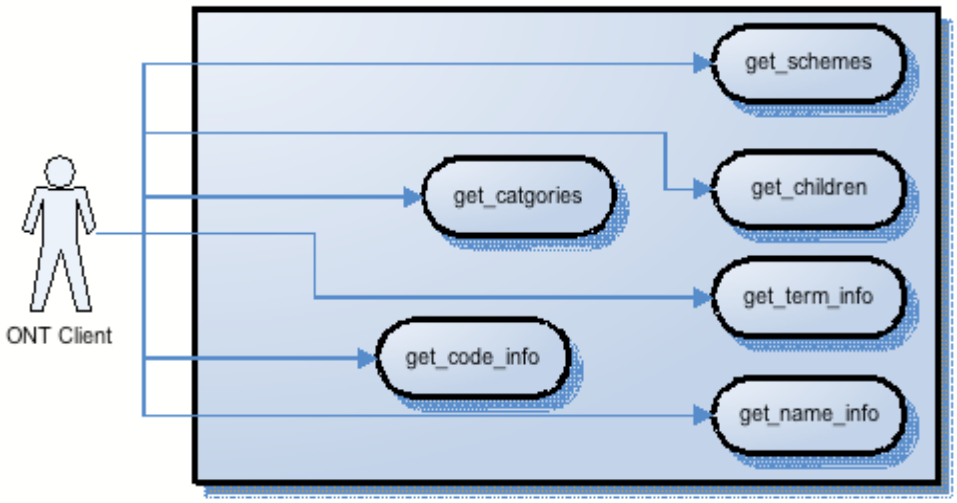
get_children: expands the top level of a vocabulary category, providing its children, for a given table and a given user.

get_schemes: returns a list of schemes available for a given user. This operation basically provides information about the different kinds of coding systems that exist.

get_name_info: returns information needed to populate a tree node for a given search keyword or name.

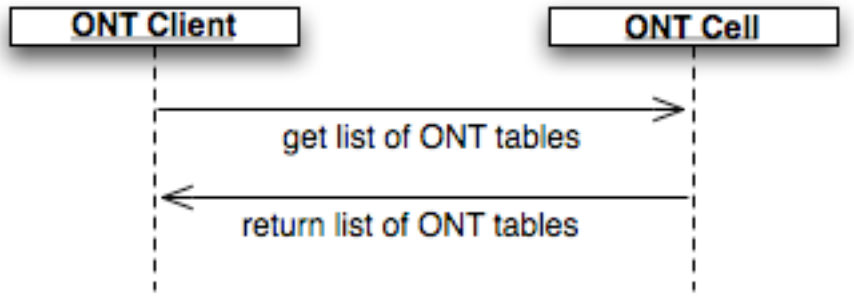
get_code_info: returns information about a code, such as the name associated with a particular code.

get_term_info: returns information about a particular child node or term.



4.1 get_categories

A get_categories message returns a role-specific list of tables that will be displayed as roots of the Navigate Terms query tree and list of categories in the Find Terms tool. No other information needs to be passed to the service. User information is provided in the message_header; roles will be provided by the Project Management (PM) cell.



4.1.1 Get a list of tables associated with a user

The getCategories message is sent by the query tool to populate the root nodes and by the find tool to populate the list of categories available to a user. It is also used internally by the other messages to help confirm that the user has table access permission.

The sequence of events is as follows: (assumes max is not an issue)

1. Client requests list of tables for a given user (get_categories)
   Request type = default for Find Terms; core for Navigate Terms

2. The ONT server performs the following steps:

   a. Get a list of roles available for this user from the PM cell – (this also serves to validate the user)

   b. Query the table of tables for the list of tables associated with this users' roles

3. The client maps the list of tables to the appropriate usage, either to Navigate Terms root node or Find Terms category list

4.1.2 get_categories request message

```
<message_body>
   <get_categories type="default" blob="false"/>
</message_body>
```

Possible "type" settings:

default => Return table name/key pairs (example: Find Terms request)

core => Return all data except system/date information (example: Navigate Terms request)

Possible "blob" settings:

false=> Do not return data stored as a blob or clob (e.g. xml, comments)

true=> Return xml and comments.

4.1.3 get_categories response message

For a Find Terms request (*type=default blob=false)*:

Response message:

```
<message_body>
  <concepts>
    <concept>
      <key>\\i2b2\RPDR\Diagnoses</key>
      <name>Diagnoses</name>
    </concept>
  </concepts>
</message_body>
```

For a Navigate Terms request (*type=core blob=true)*:

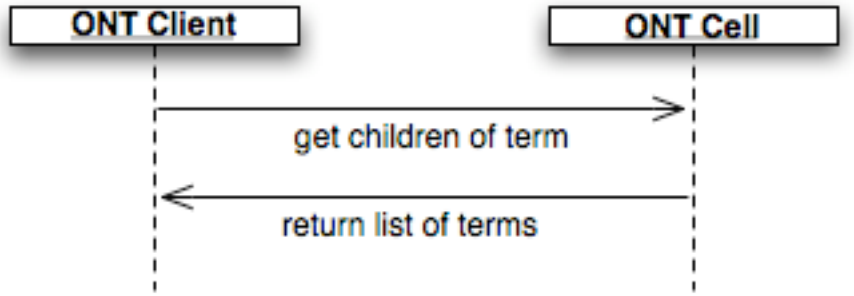
Response message:

```
<message_body>
  <concepts>
    <concept>
      <level>0</level>
      <key>\\i2b2\RPDR</key>
      <name>Ontology</name>
      <synonym_cd>N</synonym_cd>
```

```
        <visualattributes>CA </visualattributes>
        <totalnum/>
        <basecode/>
        <metadataxml/>
        <facttablecolumn>concept_cd</facttablecolumn>
        <tablename>concept_dimension</tablename>
        <columnname>concept_path</columnname>
        <columndatatype>T</columndatatype>
        <operator>LIKE</operator>
        <dimcode>\RPDR</dimcode>
        <comment/>
        <tooltip>Ontology</tooltip>
      </concept>
    </concepts>
</message_body>
```

4.2 get_children

'Get_Children' returns all the children of a particular term. A client may want a list of all children in order to expand a node of the vocabulary tree when a user is browsing through the tree.



4.2.1 Populating children of tree nodes

The getChildren message is used to populate tree nodes in the ontology Navigate Terms and Find Terms tools. In both of these cases the table/path (root) to search are known.

The sequence of events is as follows:

1. Client sends message with "max" set to 200 or higher; "type" = default

2. ONT server performs following steps:

   a. Calls getCategories to get list of tables for user. This validates user as well.

   b. Parses <parent> to obtain the table key and path. Queries table of tables to confirm that user/role can access the table that is referenced by table key passed in. This call returns the table name referenced by that key. If not, return coded error. Client receives error message with code 'TABLE_ACCESS_DENIED'.

   c. If max is set, the database is queried for that number of children associated with the parent passed in.

   d. If count < max or no max set, the database is queried for the entire list of children that meets the parent criteria

   e. Client receives list of children and populates tree.

   f. If count > max a coded error message is sent back. Client receives error message with code 'MAX_EXCEEDED'. A dialog box is displayed to ask if the user wants to see all nodes. If no – done. If yes – client sends another message with max= empty.

4.2.2 get_children request message

A get_children message implies that the user is passing a key/path for a parent and wants the children returned. The parent tag will tell the service what metadata table/path to search in and for the get_children message must be specified. The structure of parent is organized as follows: \\table_key\path. So the key (i2b2) plus the path (\RPDR\Diagnoses\Circulatory system (390-459)) is the parent : <parent>\\i2b2\RPDR\Diagnoses\Circulatory system (390-459)</parent>.

The remaining attributes provide information about the results to be returned. If the number of rows found is greater than max, then an error message will be returned in the i2b2 header. If max is left out then it is interpreted that there is no max. The hiddens and synonyms attributes tell whether to return hiddens and synonyms. By default hiddens and synonyms are false, so if they are left out it will be false. The type tells which columns to select (default/core/all) By default, type is set to default so you don't have to actually include it if you want the default set of columns returned. Each message will interpret default to be a different set of columns. get_children's default set of columns include things like key, name,

visual_attributes, tooltip... If attributes = core, then all columns except the blob and the system/date information will be returned. If attributes = all then all columns except the blob are returned. The blob attribute indicates whether or not to return the blob along with the default/core/all return columns.

```
<message_body>
     <get_children max="200" hiddens="true" synonyms="true" type="default"
blob="true">
      <parent>\\i2b2\RPDR\Diagnoses\Circulatory system (390-459)</parent>
    </get_children>
 </message_body>
```

Possible "hiddens" settings:
Some ontologies exist, but for various reasons are not displayed in the query tree

false=> Do not return data categorized as "hidden"

true=> Include data categorized as "hidden"

Possible "synonyms" settings:

Some ontologies are listed as synonyms for other ontologies.

false => Do not return data categorized as "synonym"

true => Include data categorized as "synonym"

Possible "type" settings:

default => Return all data except system/date information

core => Return all data except system/date information (same as default)

all => Return all data

Possible "blob" settings:

false => Do not return data stored as a blob or clob (xml, comments)

true => Return xml and comments.

4.2.3 get_children response message

A request where (*type=default blob=true)*

Response message:

```
<message_body>
  <concepts>
    <concept>
      <level>1</level>
      <key>\\i2b2\RPDR\Diagnoses\Circulatory system (390-459)\Acute
Rheumatic fever (390-392)</key>
      <name>Acute Rheumatic fever</name>
      <synonym_cd>N</synonym_cd>
      <visualattributes>FA </visualattributes>
      <totalnum/>
      <basecode/>
      <metadataxml/>
      <facttablecolumn>concept_cd</facttablecolumn>
      <tablename>concept_dimension</tablename>
      <columnname>concept_path</columnname>
      <columndatatype>T</columndatatype>
      <operator>LIKE</operator>
      <dimcode>\ RPDR\Diagnoses\Circulatory system (390-459)\Acute
Rheumatic fever (390-392)</dimcode>
      <comment/>
      <tooltip>Diagnoses \ Circulatory system \ Acute Rheumatic fever
      </tooltip>
    </concept>
  </concepts>
</message_body>
```

4.3 get_name_info

The get_name_info message returns information needed to populate a tree node for a given search keyword or name. This message requires the user to pass a string that is queried against the *'name'* column.

4.3.1 Generate tree nodes for a given name

To generate a list of base tree nodes associated with a given search keyword or name, the sequence of events is as follows:

1. The client requests node(s) for a given name/category. If the request message indicates all categories should be searched, loop through all known categories for that user (type = default)

2. The ONT server performs the following steps:

    a. Call getCategories to get list of tables for user. This validates user as well.

    b. Query the table of tables to confirm that user/role can access category passed in. If not, return coded error. Client receives error message with code 'TABLE_ACCESS_DENIED'.

    c. If max is set, query the database for the number of entries that meet the search criteria

    d. If count < max or no max is set, query the database for entries that meet the search criteria.

    e. The client generates a list of nodes that match search criteria

    f. If count > max send an error message back. Client receives error message with code 'MAX_EXCEEDED' and displays dialog asking if user wants to see all nodes. If no – done If yes – client sends another message with max empty.

4.3.2 get_name_info request message

This message requires the user to pass a string that is queried against the *'name'* column. The category attribute does not have to be included and if it is not, all categories the user is allowed to see will be searched.

The remaining attributes provide information about the results to be returned. If the number of rows found is greater than max, then an error message will be returned in the i2b2 header. If max is left out then it is interpreted that there is no max. The *hiddens* and synonyms attributes tell whether to return hiddens and synonyms. By default hiddens and synonyms are false, so if they are left out it will be false. The type tells which columns to select (default/core/all) By default, type is set to default. Each message will interpret default to be a different set of columns. Get_name_info's default set of columns includes items that are necessary for the visualization of the data such as *key, name, visual_attributes, tooltip*. If

attributes = core, then all columns except the blob and the system/date information will be returned. If attributes = all then all columns except the blob are returned. The blob attribute indicates whether or not to return the blob along with the default/core/all return columns

The match_str tag tells the service which string to search for. It is implied by the message get_name_info that the column to search is the name. The strategy attribute explains how the search must match, (exact, left, right, contains).

```
<message_body>
   <get_name_info category="diagnosis" max="200 hiddens="true"
synonyms="true" type="core"/>
      <match_str strategy="contains">asthma</match_str>
   </get_name_info>
</message_body>
```

Possible "hiddens" settings:

Some ontologies exist, but for various reasons are not displayed in the query tree

false=> Do not return data categorized as "hidden"

true=> Include data categorized as "hidden"

Possible "synonyms" settings:

Some ontologies are listed as synonyms for other ontologies.

false=> Do not return data categorized as "synonym"

true=> Include data categorized as "synonym"

Possible "type" settings:

default => Return name information only

core => Return all data except system/date information (same as default)

all => Return all data

Possible "blob" settings:

false=> Do not return data stored as a blob or clob (e.g. xml, comments)

true=> Return xml and comments.

Possible "strategy" settings:

contains=> Return data whose name contains the match string

exact=> Return data whose name exactly matches the match string

left=> Return data whose name starts with the match string

right=> Return data whose name ends with the match string

4.3.3 get_name_info Response message

A request where (*type=core blob=false)*

```
<message_body>
  <concepts>
    <concept>
      <level>3</level>
      <key>\\i2b2\RPDR\Medications\MUL\(LME219) respiratory
agents\(LME220) antiasthmatic combinations</key>
      <name>Antiasthmatic combinations</name>
      <synonym_cd>N</synonym_cd>
      <visualattributes>FA </visualattributes>
      <totalnum>0</totalnum>
      <facttablecolumn>concept_cd</facttablecolumn>
      <tablename>concept_dimension</tablename>
      <columnname>concept_path</columnname>
      <columndatatype>T</columndatatype>
      <operator>like</operator>
      <dimcode>\RPDR\Medications\MUL\(LME219) respiratory
agents\(LME220) antiasthmatic combinations</dimcode>
      <tooltip>medications \ respiratory agents \ antiasthmatic
combinations</tooltip>
    </concept>
  </concepts>
</message_body>
```

4.4 get_term_info

A get_term_info message implies that the user is passing a key/path for a node ("self") and wants information about that node returned.

4.4.1 Return information associated with a node

The getTermInfo message is used by timeline to obtain information associated with a node. The sequence of events is as follows:

1. Client sends message with type set
2. Ont server performs following steps:
3. Calls getCategories to get list of tables for user. This validates user as well.
4. Parses <self> to obtain table key and path. Queries table of tables to confirm that user/role can access table that is referenced by table key passed in. This call returns the table name referenced by that key. If not, return coded error.
5. Client receives error message with code TABLE_ACCESS_DENIED.
6. Query db for node that meets <self> criteria.
7. Client receives information about node

4.4.2 get_term_info request message

A get_term_info message implies that the user is passing a key/path for a node ("self") and wants information about that node returned. The attributes provide information about the results to be returned. If the number of rows found is greater than max, then an error message will be returned in the i2b2 header. If max is left out then it is interpreted that there is no max. The hiddens and synonyms tells whether to return hiddens and synonyms. By default hiddens and synonyms are false, so if they are left out it will be false. The type tells which columns to select (default/core/all) By default, type is set to default so you don't have to include it if you want the default set of columns returned. Each message

will interpret "default" to be a different set of columns. getTermInfo's default set of columns will consist of name only. If attributes = core, then all columns except the blob and the system/date information will be returned. If attributes = all then all columns except the blob are returned. The blob attribute indicates whether or not to return the blob along with the default/core/all return columns.

```
<message_body>
      <get_term_info max="200" hiddens="false" synonyms="false"
type="default" blob="true">
         <self>\\i2b2\RPDR\Diagnoses\Respiratory system (460-
519)\Chronic obstructive diseases (490-496)\(493) Asthma</self>
      </get_term_info>
   </message_body>
```

Possible "hiddens" settings:

Some ontologies exist, but for various reasons are not displayed in the query tree

false=> Do not return data categorized as "hidden"

true=> Include data categorized as "hidden"

Possible "synonyms" settings:

Some ontologies are listed as synonyms for other ontologies.

false=> Do not return data categorized as "synonym"

true=> Include data categorized as "synonym"

Possible "type" settings:

default => Return all data except system/date

core => Return all data except system/date information (same as default)

all => Return all data

Possible "blob" settings:

false=> Do not return data stored as a blob or clob (e.g. xml, comments)

true=> Return xml and comments.

4.4.3 get_term_info response message

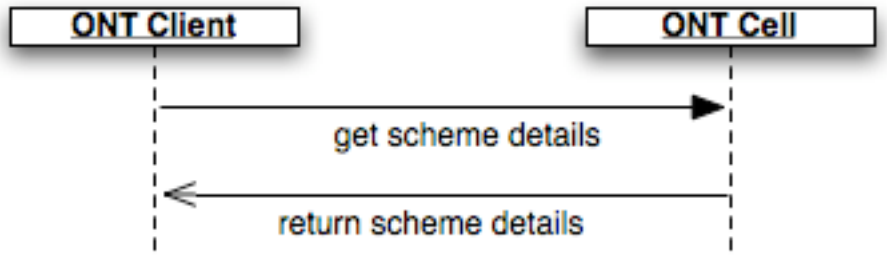
A request where (*type=default blob=true)*

```
<message_body>
  <concepts>
    <concept>
      <level>4</level>
      <key>\\i2b2\RPDR\Diagnoses\Respiratory system (460-519)\Chronic
obstructive diseases (490-496)\(493) Asthma</key>
      <name>Asthma </name>
      <synonym_cd>N</synonym_cd>
      <visualattributes>FA </visualattributes>
      <totalnum/>
      <basecode>ICD9:493</basecode>
      <metadataxml/>
      <facttablecolumn>concept_cd</facttablecolumn>
      <tablename>concept_dimension</tablename>
      <columnname>concept_path</columnname>
      <columndatatype>T</columndatatype>
      <operator>LIKE</operator>
      <dimcode>\RPDR\Diagnoses\Respiratory system (460-519)\Chronic
obstructive diseases (490-496)\(493) Asthma</dimcode>
      <comment/>
      <tooltip>Diagnoses \ Respiratory system \ Chronic obstructive
diseases \ Asthma</tooltip>
    </concept>
  </concepts>
<message_body>
```

4.5 get_schemes

'get_schemes' provides information about existing coding systems, called schemes. It returns a list of all the source systems.

This use case provides information about schemes to the client, who might want a list of all the source systems that contribute vocabulary.



A get_schemes message returns a list of schemes that will be displayed in the Ontology Find Terms tool. User information is provided in the message_header.

4.5.1 Generate scheme categories for a given user/project

To populate the list of schemes available to a user the sequence of events is as follows:

1. A client requests a list of schemes for a given user or project (getSchemes type="default")

2. The ONT server performs the following steps:

    a. Get project/role available for the user from PM cell – this also serves to validate the user.

    b. Query the table of schemes and pass back a list of schemes associated with the project/role.

3. The Client populates the scheme categories in the Find Terms tool.

4.5.2 get_schemes request message

```
<message_body>
      <get_schemes type="default"/>
   </message_body>
```

Possible "type" settings:

default => Return key/name pairs

4.5.3 get_schemes response message

Find Terms request (type=default)

Response message:

```
<message_body>
  <concepts>
    <concept>
      <key>ICD9:</key>
      <name>ICD9</name>
    </concept>
  </concepts>
</message_body>
```

4.6 get_code_info

'Get Code Info' provides information about codes. A get_code_info message implies that the user is passing a scheme:code pair and wants the associated information about that pair returned.

4.6.1 Return name/information associated with a code

To generate a list of base tree nodes or a list of names associated with a given scheme:code pair, the sequence of events is as follows:

1. A client requests information for a given scheme:code (get_code_info type= default)

2. The ONT server performs the following steps:

    a. Calls getCategories to get valid list of table data for user. This validates user as well.

    b. Queries list of tables for entries that match scheme:code pair.

3. The Client maps name/information to request.

4.6.2 get_code_info request message

A get_code_info message implies that the user is passing a scheme:code pair and wants the associated information about that pair returned. The category attribute will tell the service what metadata table to search in and does not have to be included. If category is not included, all tables the user is allowed to access will be searched.

The attributes provide information about the results to be returned. If the number of rows found is greater than max, then an error message will be returned in the i2b2 header. If max is left out then it is interpreted that there is no max. The hiddens and synonyms attributes tell us whether to return hiddens and synonyms. By default hiddens and synonyms are false, so if they are left out it will be false. The type tells which columns to select (default/core/all).By default, type is set to default so you don't have to include it if you want the default set of columns returned. Each message will interpret default to be a different set of columns. getCodeInfo's default set of columns will consist of name only. If attributes = core, then all columns except the blob and the system/date information will be returned. If attributes = all then all columns except the blob are returned. The blob attribute indicates whether or not to return the blob along with the default/core/all return columns.

```
<message_body>
     < get_code_info max="200" hiddens="true" synonyms="true"
type="default" blob="false">
          <match_str strategy="exact">ICD9:493</match_str>
       </get_code_info>
   </message_body>
```

Possible "type" settings:

default => Return name only
core => Return all data except system/date information
all => Return all data

4.6.3 get_code_info response message

For PFT request to map name to scheme:code pair (type=default blob=false)

Response message:

```
<message_body>
  <concepts>
    <concept>
      <name>Asthma</name>
    </concept>
  </concepts>
</message_body>
```

For Find Terms Search by Name request (type=core blob=false)

Response message:

```
<message_body>
  <concepts>
    <concept>
      <level>4</level>
      <key>\\i2b2\RPDR\Diagnoses\Respiratory system (460-519)\Chronic
obstructive diseases (490-496)\(493) Asthma</key>
      <name>Asthma</name>
      <synonym_cd>N</synonym_cd>
      <visualattributes>FA </visualattributes>
      <totalnum>0</totalnum>
      <basecode>ICD9:493</basecode>
      <facttablecolumn>concept_cd</facttablecolumn>
      <tablename>concept_dimension</tablename>
      <columnname>concept_path</columnname>
      <columndatatype>T</columndatatype>
      <operator>LIKE</operator>
      <dimcode>\RPDR\Diagnoses\Respiratory system (460-519)\Chronic
obstructive diseases (490-496)\(493) Asthma</dimcode>
      <tooltip>Diagnoses \ Respiratory system \ Chronic obstructive
diseases \ Asthma</tooltip>
    </concept>
  </concepts>
</message_body>
```

5. ONT Cell XML Schema Definitions

The Ontology Management XML schema consists of the following XSD files that define the <message_body> for the entire ONT cell:

ONT.xsd
Describes schema components that are common to requests and responses.

ONT_QRY.xsd
Describes the request message body for all the operations described in section 3 of this document; a query for retrieving information from rows in the metadata tables.

ONT_RESP.xsd
Describes the response message body for all the operations described in section 3 of this document; an object that holds information from rows in a metadata table.

6. Glossary

Message Tags & Attribute Definitions

For ONT _QRY.xsd:

<request>: container for request information

<get_children> container for vocabulary data request
max: the maximum number of results requested
hiddens: flag to indicate if hidden concepts should be returned
synonyms: flag to indicate if synonymous concepts should be returned
type: indicates the amount of data to be returned (default/core/all)
blob: flag to indicate if blob/clob fields should be returned

<parent> the name of the parent node at which the request is targeted

<get_term_info> container for vocabulary data request
max: the maximum number of results requested
hiddens: flag to indicate if hidden concepts should be returned
synonyms: flag to indicate if synonymous concepts should be returned
type: indicates the amount of data to be returned (default/core/all)
blob: flag to indicate if blob/clob fields should be returned

<self> the name of the node at which the request is targeted

<get_name_info> container for vocabulary data request
category: category to search in

<get_code_info> container for vocabulary data request
category: category to search in

<match_str>: stores the value of a string that should be matched.
strategy: matching strategy (exact/left/right/contains)

<get_categories> container for vocabulary data request
type: indicates the amount of data to be returned (default/core/all)
blob: flag to indicate if blob/clob fields should be returned

<get_schemes> container for vocabulary data request
type: indicates the amount of data to be returned (default/core/all)
blob: flag to indicate if blob/clob fields should be returned

For ONT_RESP.xsd:

<response> container for response information

<concepts> container for list of concepts returned in a response

<concept> container for a concept returned in a response; an object that holds information from rows in a metadata table.