3.2.2.3 CONTEXT DIAGRAM

RequestONT ServerOntology WebserviceusesRequest DAODatabase Vocabulary
Data ObjectusesencapsulatescreatesobtainsResponseRequest Handler
The Request Handler determines which Persistent Storage Location (PSL) is associated with a project request.

Sequence Diagram

Business ObjectData Access ObjectData SourceTransfer Object1. Create2. Get Data2.1. Get Data2.2. Create2.3. Return Object

Mappings of Styles

The following table is a mapping between the elements in the Component & Connector Client-Server view shown in section 3.1.1, and the Modules Decomposition and Uses views shown in sections 3.2.1 and 3.2.2.
The relationship shown is is-implemented-by, i.e. the elements from the C&C view shown at the top of the table are implemented by any selected elements from the Modules views, denoted by an "X" in the corresponding cell.

	ONT Server	**Project Management Server**	**Metadata Database**
ONT Service	X	X	
Ontology Webservice	X		
Request Handler	X		
Request DAO	X		X
Vocabulary Data Object	X		

Data View

Selecting the Data Source

Both the metadata and the patient data are distributed to projects through the existence of independent databases (in SQL Server) or schemas (in Oracle). These will be referred to in the rest of the text as the "persistent storage location" or PSL. These PSL's are organized so that the data from two metadata representations can be merged to a "Super" data set. While a person is working on a specific project, they will be directed to data in a PSL associated with that project.
In order to support the i2b2 project distribution strategy, the user is enrolled in numerous projects recorded within the i2b2 project management cell". The projects available to the user are returned in the web service call to the Project Management cell. The logic of selecting the correct PSL for the project is embodied in the following table:

DB_LOOKUP		
PK	**c_domain_id**	**VARCHAR(255)**
PK	**c_project_path**	**VARCHAR(255)**
PK	**c_owner_id**	**VARCHAR(255)**
	c_db_fullschema	**VARCHAR(255)**
	c_db_datasource	**VARCHAR(255)**
	c_db_servertype	**VARCHAR(255)**
	c_db_nicename	**VARCHAR(255)**
	c_db_tooltip	VARCHAR(255)
	c_comment	CLOB
	c_entry_date	DATE
	c_change_date	DATE
	c_status_cd	CHAR(1)

The logic for selecting the PSL is as follows:

1. There are two methods to select the correct PSL, an implicit one, and an explicit one. Both rely only on information available within the i2b2 header.

1. The implicit one relies upon the data within the <domain> tag, the <username> tag, and the <project_id> tag.
2. The explicit one relies upon the data only within the <project_id> tag. It has the format represented as the following string:

"DOMAIN"	"Project" | "USER_ID"

- ***These may not actually match the domain and username that is actually being used (since it is being built by the client), and must be checked when the PM cell is accessed.***

1. The table is meant to provide a series of default locations if ones are not specifically listed. If a project is listed in the _c_project_path_ column, then that PSL may be used, otherwise, a domain source will be used.
2. If a username is listed in the _c_owner_id_ column, then if the project also matches the *project_id*, the PSL in that row may be used, otherwise, a project PSL will be used, and if a project PSL does not exist, the domain PSL will be used. For example, only if the *domain|project|user_id* is an EXACT match to the entries in the database will that PSL be used.
3. The project may not have an entry in the table, and in that case any project would be designated the PSL of the domain.
4. If a general domain PSL is not available in the table, and only a specific project is associated with the domain in the table, then any incoming messages not associated with that project will return an error.
5. In the table, the "@" character is used to represent the absence of an entry (rather than a blank or a null).
6. In the explicit string and in the <project_id>, and "@" can be used to optionally represent a blank.

Other columns are specified as follows:

1. The column _c_db_fullschema_ is used to contain the path to a table when the data source is used. Software is written so that the absence of the delimiter (usually a ".") does not need to be explicitly stated.
2. The column _c_db_datasource_ is used to contain a short string that represents a data source configured in some other location.
3. The column _c_db_servertype_ can be "ORACLE" or "SQLSERVER".
4. The column _c_db_nicename_ is a string that can be used in client software to describe a data source.
5. The column _c_db_tooltip_ contains a longer (hierarchical) representation of the nicename.

To restate, many cells need to access some kind of persistent storage, and these cells will organize their persistent storage so that it is self contained and can be apportioned in a way consistent with the project-based requirements of i2b2 described above. To that end, a table exists in many cells to make the decision of what persistent storage location to which a specific user will be directed, depending on the project and domain to which they are associated.

Schemas within the metadata data source

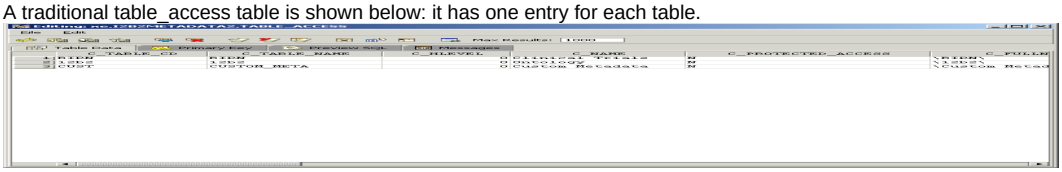
The following schemas provide data used by the ONT system:
Table_Access table

This table is used to obtain a list of root elements of the Metadata tree. Each root element of the tree is represented by a single row in this table. The primary identifier of each Metadata root element in the ***Table_Access*** table is in the "{*}c_table_cd*" column. All messages that need to point to a specific Metadata root element will use this identifier, for example in the <Key> element of many messages where this identifier is represented as *c_table_name_ (see messaging specification). Within the _c_table_name* column is the actual name in the PSL of the Metadata table. The _c_protected_access_ boolean column designates whether one must have the protected access role (*data_prot*) to obtain data from this table. The other columns are defined in a similar manner to the columns of the Metadata table, with the following special notes:
The _c_dimcode_ is used to allow the entire contents of a table to be queried in the data repository if the dimension table of the data repository has been set up in this fashion. While the _c_hlevel_ is traditionally 0 in this table (indicating one metadata table for each root element of the metadata tree), it is possible to 'flatten' the metadata tree so that multiple entries of single metadata table appear as root nodes. (see examples below)

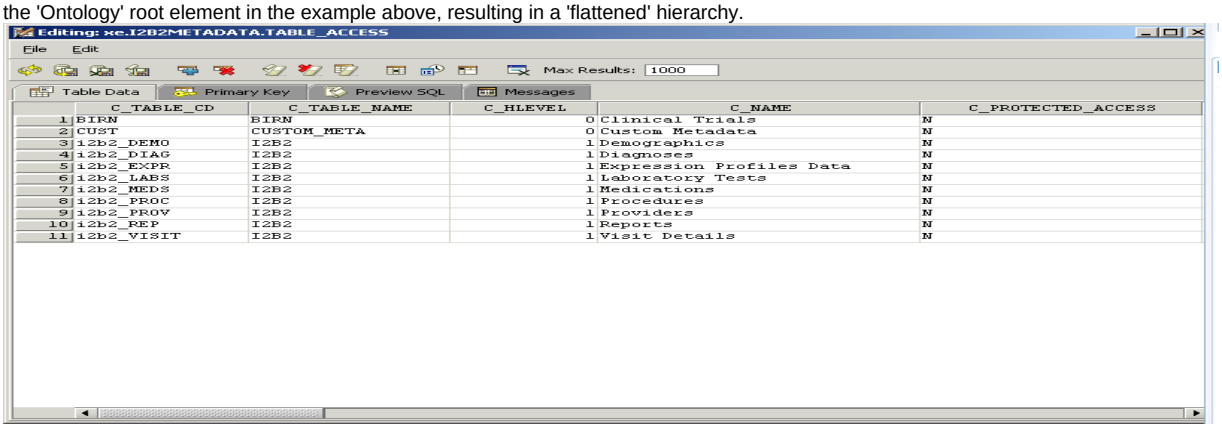
TABLE_ACCESS		
PK	**c_table_cd**	**VARCHAR(50)**
	c_table_name	**VARCHAR(50)**
	c_protected_access	CHAR(1)
	c_hlevel	**INT**
	c_name	**VARCHAR(2000)**
	c_fullname	**VARCHAR(700)**
	c_synonym_cd	**CHAR(1)**
	c_visualattributes	**CHAR(3)**
	c_tooltip	VARCHAR(900)
	c_total_num	INT
	c_basecode	VARCHAR(50)
	c_comment	CLOB

	c_metadataxml	CLOB
	c_facttablecolumn	**VARCHAR(50)**
	c_dimtablename	**VARCHAR(50)**
	c_columnname	**VARCHAR(50)**
	c_columndatatype	**VARCHAR(50)**
	c_operator	**VARCHAR(10)**
	c_dimcode	**VARCHAR(700)**
	c_entry_date	DATE
	c_change_date	DATE
	c_status_cd	CHAR(1)
	valuetype_cd	VARCHAR(50)

A traditional table_access table is shown below: it has one entry for each table.



The following table_access table shows entries for multiple root elements from a single metadata table. The nine I2B2 elements shown here appear under the 'Ontology' root element in the example above, resulting in a 'flattened' hierarchy.



	C_TABLE_CD	C_TABLE_NAME	C_HLEVEL	C_NAME	C_PROTECTED_ACCESS
1	BIRN	BIRN	0	Clinical Trials	N
2	CUST	CUSTOM_META	0	Custom Metadata	N
3	i2b2_DEMO	I2B2	1	Demographics	N
4	i2b2_DIAG	I2B2	1	Diagnoses	N
5	i2b2_EXPR	I2B2	1	Expression Profiles Data	N
6	i2b2_LABS	I2B2	1	Laboratory Tests	N
7	i2b2_MEDS	I2B2	1	Medications	N
8	i2b2_PROC	I2B2	1	Procedures	N
9	i2b2_PROV	I2B2	1	Providers	N
10	i2b2_REP	I2B2	1	Reports	N
11	i2b2_VISIT	I2B2	1	Visit Details	N

Metadata tables

The **Metadata** table encapsulates the vocabulary used in the data repository. A **term** (concept or provider) is a row from the Metadata table. It is the primary object used to pass vocabulary information to the requesting client. Typically a PSL will contain numerous Metadata tables, each with a name that indicates the domain that the vocabulary contained within represents.

METADATA		
PK	**c_fullname**	**VARCHAR(700)**
PK	**c_name**	**VARCHAR(2000)**
	c_hlevel	**INT**
	c_synonym_cd	**CHAR(1)**
	c_visualattributes	**CHAR(3)**
	c_tooltip	VARCHAR(900)
	c_total_num	INT
	c_basecode	VARCHAR(50)
	c_comment	CLOB

	c_metadataxml	CLOB
	c_facttablecolumn	**VARCHAR(50)**
	c_tablename	**VARCHAR(50)**
	c_columnname	**VARCHAR(50)**
	c_columndatatype	**VARCHAR(50)**
	c_operator	**VARCHAR(10)**
	c_dimcode	**VARCHAR(700)**
	update_date	**DATE**
	download_date	DATE
	import_date	DATE
	sourcesystem_cd	VARCHAR(50)
	valuetype_cd **m_applied_path** m_exclusion_cd c_path c_symbol	VARCHAR(50) **VARCHAR(700)** VARCHAR(25) VARCHAR(700) VARCHAR(50)

Also included in this table are modifiers associated with the terms contained within the Metadata table. The column 'm_applied_path' specifies the concept path, or c_fullname, of the concept the modifier is associated with. If a modifier applies to a concept and its descendents the 'm_applied_path' is appended with a '%'. Entries in the metadata table that are not modifiers should have the 'm_applied_path' set to '@'.
Additionally, its possible to specifiy concept(s) a modifier is not associated with. Assume a modifier has been assigned to a root concept and its descendents with an applied path of '\Diagnoses%'. To exclude this modifier from a child ('Mental disorders') of the root concept ('Diagnoses') we add an entry for the modifier with an applied path of '\Diagnoses\Mental disorders\' and set the exclusion_cd to 'X'. If we had wished to exclude this modiifer from the descendents of 'Mental disorders' we would append the applied path with a '%'.
Finally, in Release 1.6, we added two optional columns in support of external tools. The column 'c_path' contains the c_fullname of a node's parent. The column 'c_symbol' is a unique, abbreviated form of the node's c_name. A node's c_path, concatenated with its c_symbol form the node's c_fullname.

Schemes table

The Schemes schema contains the unique prefixes obtained by different source codes. For example codes from the National Drug Code are prepended with the 'NDC' prefix, while codes from the United Medical Language System are prepended with the 'UMLS' prefix. This schema contains all the schemes recognized by the ONT system.

SCHEMES		
PK	**c_key**	**VARCHAR(50)**
	c_name	**VARCHAR(5)**
	c_description	VARCHAR(100)

An example of a Schemes table is shown below:

	C.KEY	C.NAME	C.DESCRIPTION
▶ 1	NDC:	NDC	National Drug Code
2	DSG-NLP:	DSG-NLP	Natural Language Processing Data
3	UMLS:	UMLS	United Medical Language System
4	LCS-I2B2:	LCS-I2B2	Local coding system for NLP results
5	ICD9:	ICD9	ICD9 code for diagnoses and procedures
6	LOINC:	LOINC	Lab codes

Process Status table

Release 1.5 introduced the ablility to create new metadata or edit existing metadata within the i2b2 workbench. Once metadata has been created or edited it is necessary to synchronize these changes with the appropriate dimension table. The concept_dimension table would need to be updated with new concept terms; likewise provider_dimension table would need to be updated with new provider terms and the modifier_dimension table could need to be updated with new modifier terms.
The Process Status table provides information about the synchronization between the ontology metadata tables and the dimension tables.

PROCESS STATUS		

PK	process_id	INT
	process_type_cd	VARCHAR(50)
	start_date	DATE
	end_date	DATE
	process_step_cd	VARCHAR(50)
	process_status_cd	VARCHAR(50)
	crc_upload_id	VARCHAR(5)
	status_cd	VARCHAR(50)
	message	VARCHAR(2000)
	entry_date	DATE
	change_date	DATE
	changedby_char	CHAR(50)

Deployment View

Global Overview

ONT ClientInternetWebServerJ2EE ApplicationServerDatabase ServerWebservicePM Service

Detailed deployment model

ontology.aarontology-server.jarontology-core.jar<<deployment>><<deployment>><<deployment>>**JBoss Application ServerOracleDatabaseServer**