Informatics for Integrating Biology and the Bedside

i2b2 Software Architecture

Identity Management Framework (IM) Cell

Document Version: *1.7.0*

i2b2 Software Version: *1.7.00*

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
1.7.1	10/23/13	Mike Mendis	Updated Jboss and Axis2

ABSTRACT

This is a software architecture document for the **Identity Management Framework (IM) cell**. It identifies and explains the important architectural elements. This document will serve the needs of stake holders to understand the system concepts and give a brief summary of the use of the IM message format.

1. OVERVIEW

The Identity Management Framework cell (IM) is one of the core cells in the i2b2 Hive. The cell manages

1.1. IM Definitions, Acronyms and Abbreviations

1.1.1. IM Data Object (IMDO)

This object holds patient and site information and performs auditing.

1.2. Roles

When and how data is presented to a user is based on their user roles, which are specified in the PM Cell. Each user will have at least two roles per user_ID and product_ID combination. These two roles can be further defined as a *Data Protection role* and a *Hive Management role*.

The data protection role establishes the detail of data the user can see while the hive management role defines the level of functionality the user has in a project. The following tables summarize the roles in a hierarchical order of least to most access.

Data Protection Track	
Role	**Access Description**
DATA_OBFSC	OBFSC = Obfuscated ▪ The user can see aggregated results that are obfuscated (example: patient count). ▪ The user is limited on the number of times they can run the same query within a specified time period. If the user exceeds the maximum number of times then their account will be locked and only the Admin user can unlock it.
DATA_AGG	AGG = Aggregated ▪ The user can see aggregated results like the patient count. ▪ The results are not obfuscated and the user is not limited to the number of times they can run the same query.
DATA_LDS	LDS = Limited Data Set ▪ The user can see all fields except for those that are encrypted. ▪ An example of an encrypted field is the *blob fields* in the *fact* and *dimension tables*.
DATA_DEID	DEID = De-identified Data

	▪ The user can see all fields including those that are encrypted. ▪ An example of an encrypted field is the *blob fields* in the *fact* and *dimension tables*.
DATA_PROT	PROT = Protected ▪ The user can see all data, including the identified data that resides in the Identity Management Cell.

Hive Management Track	
Role	**Access Description**
USER	Can create queries and access them if he / she is the owner of the query.
MANAGER	Can create queries as well as access queries created by different users within the project

Further details regarding roles can be found in the PM_Design_Document.

1.3. Security

Users can access the IM with user-id and password combination, which is authenticated through the Project Management Cell. The implementation detail of the Project Management Cell is considered out-of-scope to this system context.

Further details regarding the implementation of the Project Management cell can be found in the PM_Install_Guide.

1.4. Scope of the system

Some other participants, currently outside the scope of the IM are:

- Project Management Cell
- Data Repository (CRC) Cell

1.5. Assumptions / Constraints

- The Identity Management database will contain **protected health information**.

1.6. Technical Platform

The technology used to build the product is as follows:

- Java 2 Standard Edition 7.0
- Oracle Server 10g/11g database
- SQL Server 2005/2008
- Xerces2 XML parser
- Jboss Application server version 7.1.1
- Spring Web Framework 2.0
- Axis2 1.6.2 web service (SOAP / REST)

1.6.1. Transaction

The IM system is transactional, leveraging the transaction management model of the J2EE platform.

1.6.2. Security

The application must implement basic security behaviors:

Category	Behavior
Authentication	Authenticate using at least user name and a password.
Authorization	Based on the user role, the user may only access those categories allowed to by role.
Confidentiality	Sensitive data must be encrypted.
Data Integrity	Data sent across the network cannot be modified by a tier.
Auditing	All data received will be logged in the audit table.

1.6.3. Persistence

This application utilizes JDBC calls to retrieve persisted data.

1.6.4. Reliability / Availability

- The reliability / availability will be addressed through the J2EE platform
- Targeted availability is 16 / 7: 16 hours a day, 7 days a week
- The remaining time (8 hours) is reserved for any maintenance activities

1.6.5. Performance

- The user authentication with the project management cell must be under 1 second.

2. USE CASE

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

2.1. 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.

3. ARCHITECTURE DESCRIPTION

This document provides the description of the architecture as multiple views. Each view conveys the different attributes of the architecture.

1. Components and Connector View
    a. Client-Server Style

2. Module View
    a. Decomposition Style
    b. Uses Style

3. Data View
4. Deployment View

3.1. Components and Connector View

A **Component and Connector view** (C&C) represents the runtime instances and the protocols of connection between the instances. The connectors represent the properties such as concurrency, protocols and information flows. The diagram shown in the *Primary Presentation* section represents the Component and Connector view for the multi-user installation. As seen in the diagram, component instances are shown in more detail with specific connectors drawn in different notations.

3.1.1. Client-Server View

The IM system is represented using the C&C Client-Server view.

3.1.1.1. PRIMARY PRESENTATION

3.1.1.2. ELEMENT CATALOG

3.1.1.2.1. Elements and their Properties

The properties of IM cell elements are:

- *Element Name:* listed in the table shown below.
- *Type:* whether the element is a data repository, a data accessory, a communication method, a query, a client or a server component.
- A *description* of the element

Element Name	Type	Description
I2b2 Workbench	Client Component	Webservice client (i2b2 Workbench / Navigator) submits the requests to IM Server components and renders response XML.
IM Framework Server	Server Component	Provides Web Service Interface for the IM system. It supports REST protocols. It uses Project Management server to handle user authentication. It uses the Data Repository for patient mapping.
Project Management Server	Server Component	IM cell uses the Project Management cell to authenticate the user. The IM cell constructs the PM request message and makes a web service call to the Project Management Cell.
IM	Data Repository Component	This repository is a database for the i2b2 IM data.
JDBC	Query Connector	SQL query used as a connector between the IM System and the identity database.
Web Service	Request Connector	REST protocol used to communicate with the external system.

3.1.1.3. DESIGN RATIONALE, CONSTRAINTS

N-tier Architecture

The client-server style depicts the n-tier architecture that separates presentation layer from business logic and data access layer; thus providing for a high degree of portability through the application of the principle of Separation of Concerns.

3.2. Module View type

The module view shows how the system is decomposed into implementation units and how the functionality is allocated to these units. The layers show how modules are encapsulated and structured. The layers represent the “allowed-to-use” relation.

The following sections describe the module view using Decomposition and Uses Style.

3.2.1. Decomposition Style

The Decomposition style presents the functionality in terms of manageable work pieces. They can be further decomposed to present higher level of details. The decomposition view identifies modules and breaks them down into sub-modules and so on, until a desired level of granularity is achieved. The “Uses” style shows the relationships between modules and sub-modules. This view is very helpful for implementation, integration and testing the system.

3.2.1.1. PRIMARY PRESENTATION

System	Segment
IM Framework Server	Operation Manager

3.2.1.2. ELEMENT CATALOG

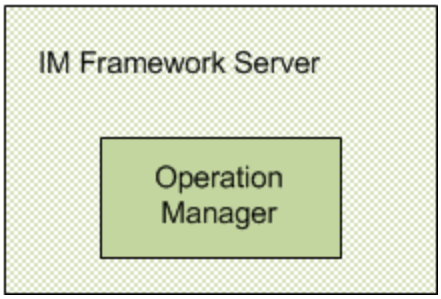
3.2.1.2.1. Elements and their properties

Element Name	Type	Description
Operation Manager	Subsystem	This subsystem manages queries for IM operations

3.2.1.3. RELATIONS AND THEIR PROPERTIES

The subsystem elements form the ***is-part*** of the relation with the overall IM system.

3.2.1.4. CONTEXT DIAGRAM



3.2.2. Uses Style

The Uses style shows the relationship between modules and sub-modules. This view is very helpful for implementing, integrating and testing the system.

3.2.2.1. PRIMARY PRESENTATION

System	Segment
IM Framework Server	IM Module
Operation Manager Subsystem	IM Web Service
	Request Handler
	Request DAO
	Patient Data Object or Audit

3.2.2.2. ELEMENT CATALOG

3.2.2.2.1. Elements and their Properties

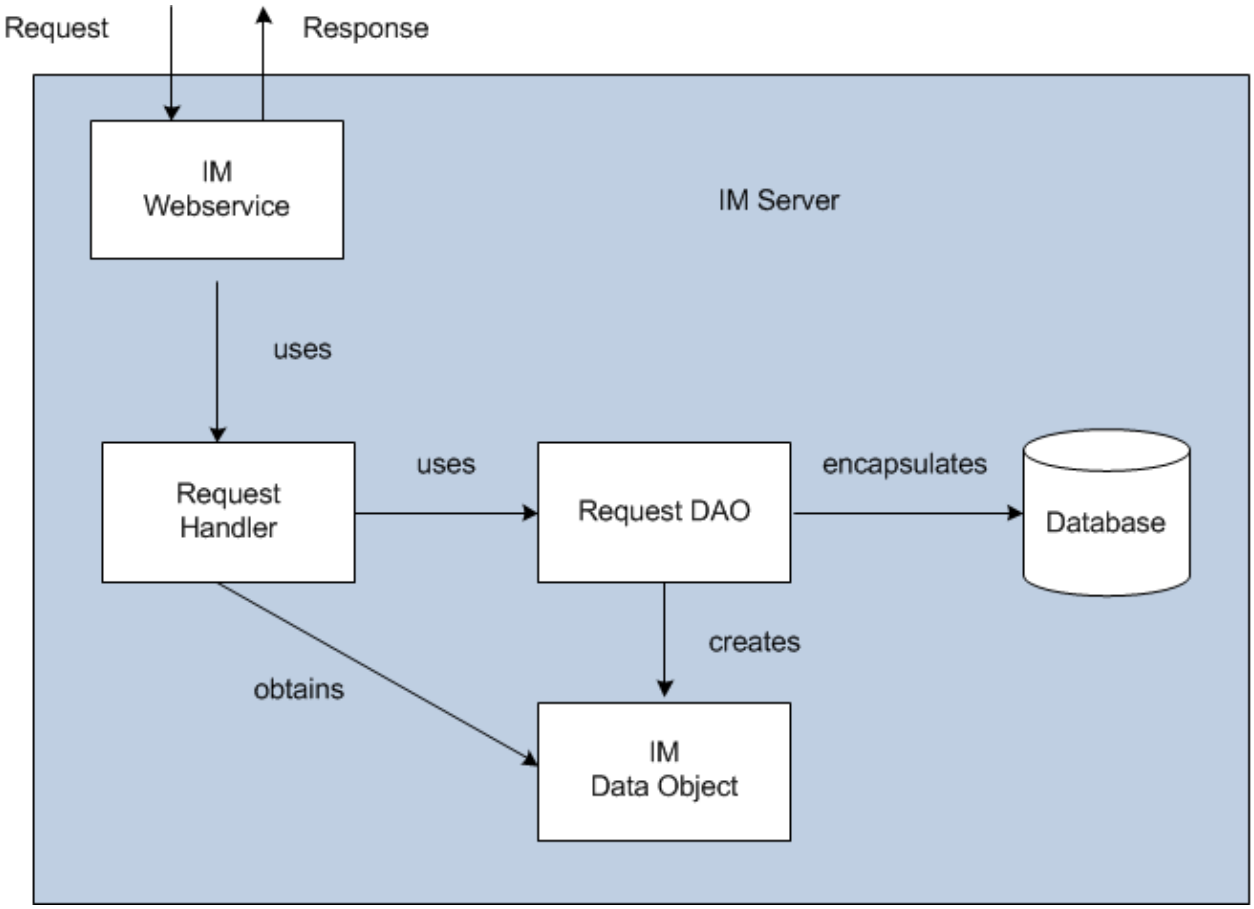
Element Name	Type	Description
IM Module	Module	User Login Module authenticates through PM Server System.
IM Webservice	Communication Module	Provides web service interface to IM operations.
Request Handler	Business Object	Delegates IM requests to Data Access Object layer to perform database operations.
Request DAO	Data Access	Supports database query operations.

	Object	
Patient Data Object	Transfer Object	Object representation of persisted data.

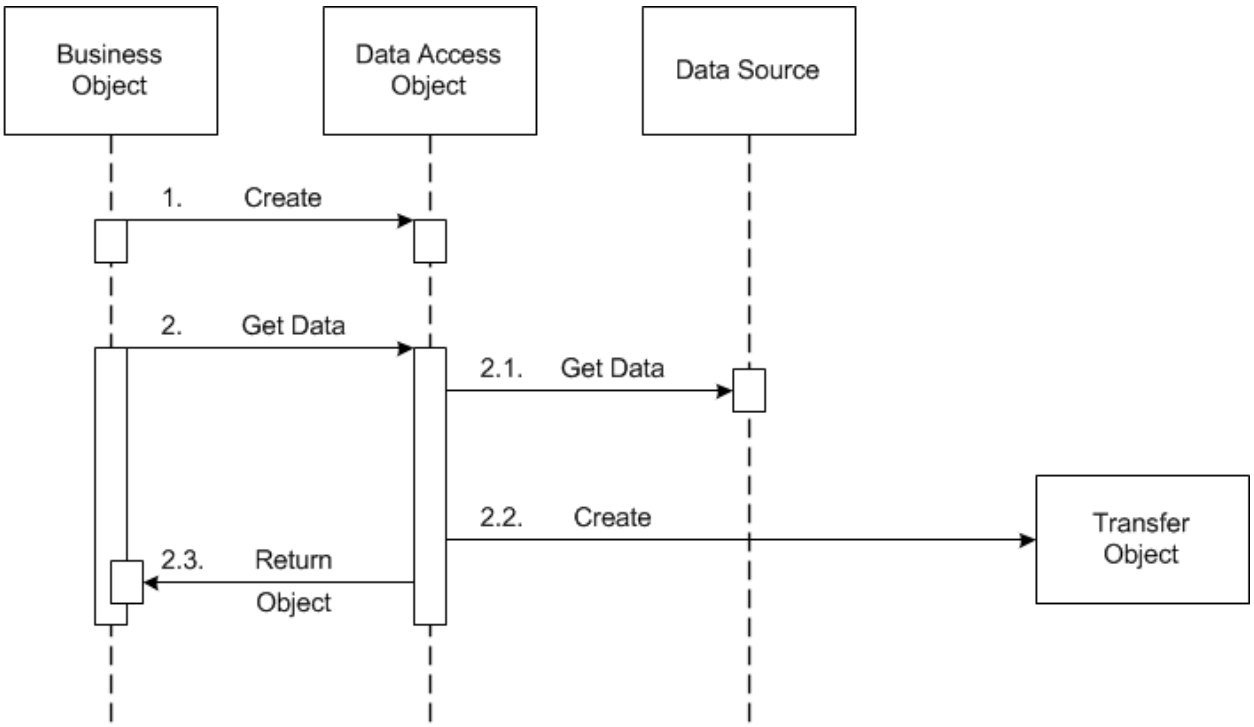
3.2.2.3. RELATION AND THEIR PROPERTIES

The modules in this style follow a ***depends-on*** relation.

3.2.2.4. CONTEXT DIAGRAM



3.2.2.5. SEQUENCE DIAGRAM



3.3. 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 Uses view* and *Decomposition view* 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 and “X” in the corresponding cell.

	IM Server	**Data Repository Server**	**PM Server**	**IM Database**
IM Service	X	X	X	
IM Webservice	X			
Request Handler	X			
Request DAO	X			X
Patient Data Object	X			

4. DATA VIEW

4.1. Selecting the Data Source

Stored Workplace data is 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 document 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 project 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.

a. The implicit one relies upon the data within the <domain> tag, the <username> tag, and the <project_id> tag.

b. The explicit one relies upon the data only within the <project_id> tag. It has the format represented as the following string:

|"DOMAIN" | "PROJECT" \ "sub-project" \ "sub-sub-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.

2. 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.
3. If a username is listed in the *C_OWNER_ID* column, and the project also matches the *PROJECT_ID*, the PSL in that row may be used otherwise a project PSL will be used. If the 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.

4. The project id may have associated sub-projects that will be represented as *project* \ *sub-project* \ *sub-sub-project* string. If a sub-project is identified but only the project exists in the table then the project PSL would be used.
5. The project may not have an entry in the table and in that case any project (and sub-projects) would be designated the PSL of the domain.
6. 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.
7. In the table, the "@" character is used to represent the absence of an entry (rather than a blank or a null).
8. In the explicit string and in the <project_id> an "@" can be used to optionally represent a blank column.

Other columns are specified as follows:

9. 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.
10. The column *C_DB_DATASOURCE* is used to contain a short string that represents a data source configured in some other location.
11. The column *C_DB_SERVERTYPE* can be "ORACLE" or "SQLSERVER".
12. The column *C_DB_NICENAME* is a string that can be used in the client software to describe a data source.
13. 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 that are 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.

4.2. Schemas within the IM Data Source

The following schemas provide data used by the IM system:

4.2.1. General Information

All the tables have the following five technically-oriented or administrative columns, except for the audit table.

Column Name	Data Type	Nullable	Definition
UPDATE_DATE	datetime	Yes	Date the row was update by the source system The date is obtained from the source system
DOWNLOAD_DATE	datetime	Yes	Date the data was downloaded from the source system
IMPORT_DATE	datetime	Yes	Date the data was imported into the CRC
SOURCESYSTEM_CD	datetime	Yes	A coded value for the data source system
UPLOAD_ID	datetime	Yes	A numeric id given to the upload

4.2.2. Demographics Table

The **IM_MPI_DEMOGRAPHICS** table contains the demographics data associated with the site id; this information can be populated from EMR systems.

Columns with an * are optional

IM_MPI_DEMOGRAPHICS		
PK	**GLOBAL_ID**	**VARCHAR(50)**
	GLOBAL_STATUS	VARCHAR(50)
*	FIRST_NAME	VARCHAR(50)
*	LAST_NAME	VARCHAR(100)
*	ADDRESS_1	VARCHAR(100)
*	ADDRESS_2	VARCHAR(100)
*	CITY	VARCHAR(50)
*	STATE	VARCHAR(50)
*	ZIP	VARCHAR(20)
*	COUNTRY	VARCHAR(100)
	UPDATE_DATE	DATETIME
	DOWNLOAD_DATE	DATETIME
	IMPORT_DATE	DATETIME
	SOURCESYSTEM_CD	VARCHAR(50)
	UPLOAD_ID	INT

4.2.3. Mapping Table

The **IM_MPI_MAPPING** table links the global id with the site data. All patient ids in the LCL_ID column are unencrypted and LCL_SITE do not end in a '_E', where '_E' symbolizes that the site is encrypted. If your UPDATE_DATE does not include the time than in order to guarantee that either a sequence or IMPORT_DATE should be added to the primary key.

IM_MPI_MAPPING		
	GLOBAL_ID	**VARCHAR(200)**
PK	**LCL_SITE**	**VARCHAR(50)**
PK	**LCL_ID**	**VARCHAR(200)**
PK	**UPDATE_DATE**	**DATETIME**
	LCL_STATUS	**VARCHAR(50)**
	DOWNLOAD_DATE	DATETIME
	IMPORT_DATE	DATETIME
	SOURCESYSTEM_CD	VARCHAR(50)
	UPLOAD_ID	INT

4.2.4. Project Sites Table

The **IM_PROJECT_SITES** table links a project with an associated site.

IM_PROJECT_SITES		
PK	**PROJECT_ID**	**VARCHAR(50)**
PK	**LCL_SITE**	**VARCHAR(50)**
	PROJECT_STATUS	VARCHAR(50)
	UPDATE_DATE	DATETIME
	DOWNLOAD_DATE	DATETIME
	IMPORT_DATE	DATETIME
	SOURCESYSTEM_CD	VARCHAR(50)
	UPLOAD_ID	INT

4.2.5. Project Patient Table

The **IM_PROJECT_PATIENTS** table links project with site.

IM_PROJECT_SITES		
PK	**PROJECT_ID**	**VARCHAR(50)**
PK	**GLOBAL_ID**	**VARCHAR(200)**
	PATIENT_PROJECT_STATUS	VARCHAR(50)
	UPDATE_DATE	DATETIME
	DOWNLOAD_DATE	DATETIME
	IMPORT_DATE	DATETIME
	SOURCESYSTEM_CD	VARCHAR(50)
	UPLOAD_ID	INT

4.2.6. Audit Table

The **AUDIT** table links project with site.

IM_PROJECT_SITES		
PK	**QUERY_DATE**	**DATETIME**
	LCL_SITE	VARCHAR(50)
	LCL_ID	VARCHAR(200)
	USER_ID	VARCHAR(50)
	PROJECT_ID	VARCHAR(50)
	COMMENTS	TEXT

5. EMPI VIEW

5.1. Overview

The IM cell can use a third party EMPI system to get more information on the patient.

5.2. EMPI Interface

Create a new class that extends the EMPI interface located in edu.harvard.i2b2.im.util. Three methods will need to be implemented.

findPerson - this is the public facing method that returns the xml data about the person.

parse – Will extract the patient parameters from the empi service and turn a parameter array

getIds – Will return a list of medical record numbers for that specific patient

The EMPIOpenEMPI.java shows and example of how i2b2 connects to this EMPI service and extracts the data/

REFERENCES

Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R. and Stafford, J., *Documenting Software Architectures: Views and Beyond*. (Boston, MA: Addison-Wesley, 2003)

Philippe Kruchten, "Architectural Blueprints – The "4+1" View Model of Software Architecture, *http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf* (*IEEE Software 12 (6)*, November 1996)

"Object Management Group UML 2.0 Specification", *http://www.omg.org/technology/documents/formal/uml.htm* (Object Management Group)

i2b2 (Informatics for Integrating Biology and the Bedside) *https://www.i2b2.og/resrcs/hive.html*