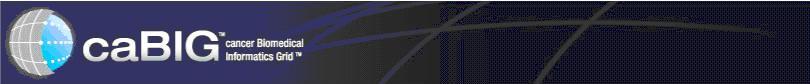
***Technical Architecture Guide***

**caIntegrator2**

**DOCUMENT CHANGE HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Version Number | Date | Contributor | Description |
| 1.0 | 5/29/2008 | Michael Holck | Original |
| 1.1 | 1/23/2009 | Eric Tavela | Updated with current architecture and projected Grid interface |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Updated January 26, 2009



**TABLE OF CONTENTS**

1 Introduction 1

1.1 Chapter Organization and Content 1

1.2 Overview of the Guide 1

1.3 Organization of the Guide 2

1.4 Intended Audience for the Guide 2

1.5 Scope of the Guide 2

1.6 Document Text Conventions 3

1.7 Related documents 4

2 Overview of the Software 5

2.1 Chapter Organization and Content 5

2.2 Software Overview 5

2.2.1 Project Background 5

2.2.2 System Overview 5

2.2.3 Features of caIntegrator2 6

3 Architecture 7

3.1 Chapter Organization and Content 7

3.2 System Architecture Overview 7

3.3 Architectural Design 8

4 Design 11

4.1 Chapter Organization and Content 11

4.2 Study Management Service Design 11

4.2.1 Use Case 11

4.2.2 Deploy Study Process Flow 12

4.2.3 Deploy Study Sequence Diagram 12

4.2.4 Import Clinical Data Process Flow 14

4.2.5 Import Clinical Data Sequence Diagram 14

4.2.6 Configure Genomic Data Process Flow 16

4.2.7 Configure Genomic Data Sequence Diagram 16

4.2.8 Configure Imaging Data Process Flow 16

4.2.9 Configure Imaging Data Sequence Diagram 17

4.3 Technology Stack 19

5 Security and Security Management 20

5.1 Chapter Organization and Content 20

5.2 Software Authentication 20

5.3 Software Authorization 20

6 Information Model 21

6.1 Chapter Organization and Content 21

6.2 caIntegrator2 Model Development 21

6.3 caIntegrator2 Modeling Artifacts 21

6.4 caIntegrator2 Object Model 21

7 Analysis Service Interface API 26

7.1 Chapter Organization and Content 26

7.2 caIntegrator2 Service API 26

Appendix A References 29

Technical Manuals/Articles 29

caBIG Material 30

caCORE Material 30

Appendix B Glossary 31

Index 33

# Introduction

The caIntegrator line of applications has been very successful for their innovation and integration of clinical data with genomic data. Due to this success, many centers are requesting a caIntegrator-like application for their own studies. The goal of the caIntegrator 2 project is to provide the user with an application that will allow those with a somewhat moderate technical background to create a caIntegrator-like application that they can deploy at their local center. This application would acquire the data and other information from the user using caBIG APIs and/or grid services, along with CSV files, then build, configure and install a custom caIntegrator web application tailored specifically for that user, based on the data that they have supplied.

The first release of caIntegrator2 that is documented in this architecture guide will be handling clinical data using a dynamic data approach allowing the users to load clinical data from a CSV file, array data using the caArray grid services, and image data using the NCIA grid services.

## Chapter Organization and Content

The purpose of this chapter is to provide a high level overview of the caIntegrator2 solution, its components and the overall content of this guide.

Topics in this chapter include:

* Intended Audience for the Guide
* Document Text Conventions

## Overview of the Guide

This guide introduces the technical and architectural aspects of the components that make up caIntegrator2. It provides an overview of the architecture, describes the components and their use cases, identifies the technology stack and provides some technical reference.

## Organization of the Guide

This table provides a summary of the focus of each chapter.

Table - Main Contents of the caIntegrator2 Technical Architecture Guide

|  |  |
| --- | --- |
| Chapter in caXchange Technical Architecture Guide | Chapter Contents |
| Chapter 1 |  |
| Chapter 2 |  |
| Chapter 3 | **Error! Reference source not found.** |
| Chapter 4 | **Error! Reference source not found.** |
| Chapter 5 |  |
| Chapter 6 | Design |
| Chapter 7 |  |
| Chapter 8 |  |
| Appendix A | [References](#References) |
| Appendix B | [Glossary](#Glossary) |
| Index | [Index](#Index) |

## 

## Intended Audience for the Guide

The intended audience for this document includes developers and architects interested in obtaining an overview of the architecture and high level design for the caIntegrator2 application.

## Scope of the Guide

The scope of this document is limited to the requirements as identified in the 1.0 version release of the caIntegrator2 application. For additional project and requirement-specific information that were used to scope this release, please refer to the following documents:

* Use Case Specifications
  + Browse Study
  + Deploy Study
  + Manage Queries
  + Perform Data Analysis
  + Review Image Data
  + Use Workspace
* Design model

The latest versions of these documents can be found in the:

[caIntegrator2 GForge Documents Page](https://gforge.nci.nih.gov/docman/?group_id=507)<https://gforge.nci.nih.gov/plugins/scmcvs/cvsweb.php/caxchange/Deliverables/?cvsroot=caxchange>

## Document Text Conventions

The following table shows various typefaces to differentiate between regular text and menu commands, keyboard keys, and text that you type. This illustrates how conventions are represented in this guide.

Table - Document Conventions

| Convention | Description | Example |
| --- | --- | --- |
| **Bold & Capitalized Command**  **Capitalized command > Capitalized command** | Indicates a Menu command  Indicates Sequential Menu commands | **Admin > Refresh** |
| text in small caps | Keyboard key that you press | Press enter |
| text in small caps + text in small caps | Keyboard keys that you press simultaneously | Press shift + ctrl and then release both. |
| Special typestyle | Used for filenames, directory names, commands, file listings, source code examples and anything that would appear in a Java program, such as methods, variables, and classes. | URL\_definition ::= url\_string |
| **Boldface type** | Options that you select in dialog boxes or drop-down menus. Buttons or icons that you click. | In the Open dialog box, select the file and click the **Open** button. |
| *Italics* | Used to reference other documents, sections, figures, and tables. | *caCORE Software Development Kit 1.0 Programmer’s Guide* |
| ***Italic boldface type*** | Text that you type | In the New Subset text box, enter ***Proprietary Proteins.*** |
| Note: | Highlights a concept of particular interest | **Note:** This concept is used throughout the installation manual. |
| **Warning!** | Highlights information of which you should be particularly aware. | **Warning!** Deleting an object will permanently delete it from the database. |
| {} | Curly brackets are used for replaceable items. | Replace {root directory} with its proper value such as c:\cabio |

## Related documents

For additional project information, please refer to the following documents:

* Software Requirements Specification (SRS) (TBD)
* Use Case Specification and Realization Documents
* Design Model
* Installation Guide (TBD)
* End User’s Guide (TBD)

The latest versions of these documents can be found in the:

[caIntegrator2 GForge Documents Page](https://gforge.nci.nih.gov/docman/?group_id=507)

# Overview of the Software

## Chapter Organization and Content

Topics in this chapter include:

* **Error! Reference source not found.**

## Software Overview

### Project Background

There have been several caIntegrator projects developed over the last few years that have been very successful for the specific studies they were built for. However, in doing them we have learned that the data that they want to include and query continues to grow and change throughout the life of the project. Previously this always required data model changes and custom development to include these new attributes. The goal for this program was to develop a system that could handle this dynamic data problem and still leverage the work that has already been done for other caIntegrator programs. In addition, there has been a great deal of work done for storing array data (caArray), and imaging data (NCIA) that could be consumed for use in this type of an application.

### System Overview

The components detailed in this technical architecture guide deal with the handling of dynamic clinical data coming from CSV files, array data coming from the caArray grid services, and imaging data coming from the NCIA grid services. There are also components to handle the concept of user workspaces and managed queries by user. This first release of caIntegrator2 will focus on handling a scenario where we allow the user to setup a study that includes clinical, array, and imaging data and perform queries across all three data types and perform some simple analysis on this data.

The key roles in the project are defined in the following table.

Table - Key Roles in the caXchange Project

|  |  |  |
| --- | --- | --- |
| Role | Name | Comments |
| Project Sponsor | ICR, caBIG |  |
| Reviewer | caBIG Architecture Workspace | Will review and provide guidance of architecture artifacts |
| User | TBD | Will provide UI and usability feedback and recommendations |
| Developer | ScenPro, 5AM, SAIC | Joint development team |

### Features of caIntegrator2

Key features of the caIntegrator2 application include the following:

* Supports user authentication and authorization
* Provides caCORE SDK generated caIntegrator2 model object APIs
* Allows dynamic clinical data definition
* Accesses array data from caArray
* Accesses imaging data from NCIA
* Supports user workspace and saved queries

# Architecture

## Chapter Organization and Content

Topics in this chapter include:

* System Architecture Overview
* Architectural Design

## System Architecture Overview

The following diagram depicts the caIntegrator2 high level architecture from a data centric point of view.



Figure – caIntegrator2 High level Architecture

The three types of data handled by the first release are shown on the left in their input state. They are consumed by the caIntegrator2 system and stored in either the database or a high-throughput data file. Then this data is queried, browsed, or analyzed by the user.

## Architectural Design

This technical architecture guide covers the major components that are a part of the caIntegrator2 first release effort. The following diagram depicts the components of the caIntegrator2 application:



Figure – caIntegrator2 Component Model

The following table describes the components or subsystems that make up the caIntegrator2 application:

Table - Summary of caIntegrator2 Subsystems

| **Subsystem** | **Description** | **Req’ts Addressed** |
| --- | --- | --- |
| caIntegrator2 Data Service | The caIntegrator2 Data Service is a caBIG compatible grid service that will provide the data stored in the caIntegrator2 database |  |
| Study Management Service | The Study Management Service will provide the capability to create and deploy a new study which will allow the user to specify the data to be used for that study and its visibility. It will also manage the updating of any study that was previously created. |  |
| Workspace Service | The Workspace service will allow the user to setup a list of studies, and queries that they want to save and reuse. |  |
| Analysis Management Service | The Analysis Management Service will package the data to be analyzed and manage the handing off to the analysis server and then manage the retrieval of the analysis for display to the user. |  |
| Query Management Service | The Query Management Service will allow users to construct queries for all data types and store these queries for later reuse |  |
| Array Data Service | The Array Data Service will manage the acquisition and storage of all array data from the caArray grid service. |  |
| Image Data Service | The Image Data Service will manage the acquisition and storage of all image data from the NCIA grid service |  |
| caIntegrator2 DAO | The caIntegrator2 DAO will handle using the caCORE generated APIs for retrieval of data and will also provide the insert methods to create and update data in the database. |  |
| caArray Façade | The caArray Façade is strictly an interface for accessing the caArray grid service and is used by the Array Data Service for querying and retrieving data from caArray. |  |
| NCIA Façade | The NCIA Façade is strictly an interface for accessing the NCIA grid service and is used by the Image Data Service for querying and retrieving data from NCIA. |  |
| caDSR Facade | The caDSR Façade is strictly an interface for querying the caDSR repository to search for Data Elements that match the annotations the user is using for clinical, image, or genomic annotations. This is used by the Study Management Service when a new study is created or additional annotations are added to a study. |  |

# Design

## Chapter Organization and Content

Topics in this chapter include:

* Study Management Service Design
* Technology Stack

## Study Management Service Design

The Study Management Service handles the deployment and updating of a study in the caIntegrator2 application.

### Use Case

The high level use case for the Study Management Service is shown below.



Figure - Study Management Service Use Case

### Deploy Study Process Flow

The study manager creates and deploys a new study in the following way:

* The Study manager selects that they want to create a new study
* The system prompts for the initial study creation information and allows the user to associate different data types to the study
* The study manager chooses to add clinical data to the study and is taken to the import clinical data use case
* (Optional) The study manager chooses to add genomic data to the study and is taken to the configure genomic data use case
* (Optional) The study manager chooses to add image data to the study and is taken to the configure imaging data use case
* The study manager selects which annotations are displayed during the browse study use case
* The study manager completes the definition of the study and chooses to deploy the study
* The system makes the study available
* (Optional) The study manager chooses to define shared queries for this study and is taken to the manage queries use case
* (Optional) The study manager chooses to define shared data lists for this study and is taken to the manage data lists use case

### Deploy Study Sequence Diagram

The sequence diagram below shows the basic sequence for creating and deploying a study using the Study Management Service.

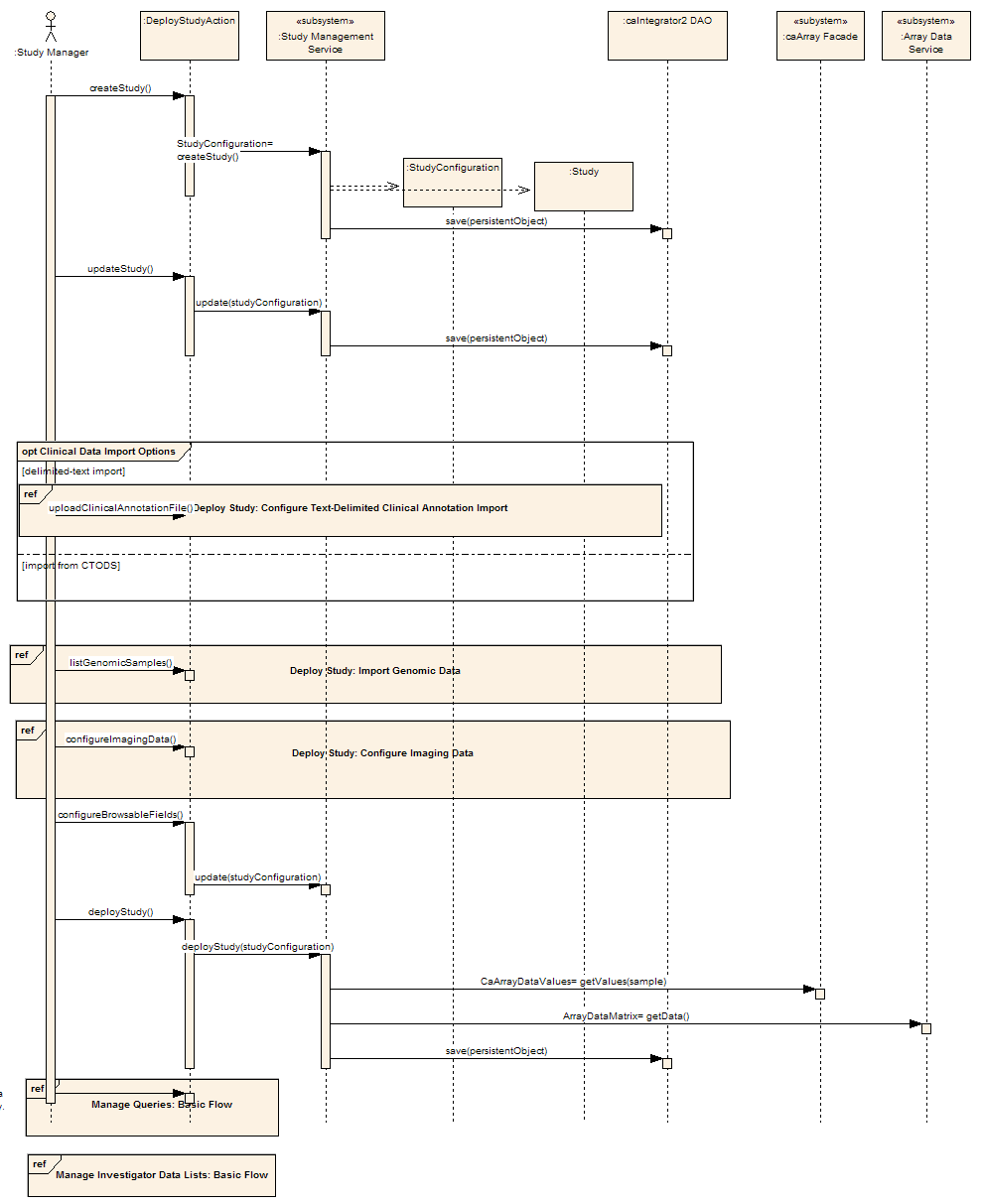


Figure – Study Management Service Deploy Study Sequence Diagram

### Import Clinical Data Process Flow

The study manager imports clinical data in the following way:

* The Study manager uploads a clinical annotations file in CSV format
* The system validates the file and begins the creation of the first subject annotation (loop)
  + The system searches for the annotation in the caIntegrator2 database for reuse
  + (Optional) The system searches for the annotation as a Data Element in caDSR for reuse and stores the caDSR public ID and version
  + (Optional) No matches were found so the Study Manager provides a definition of this annotation and if applicable a list of permissible values
  + The system stores this annotation definition in the database
* The system then iterates through the clinical data file creating subjects and setting their annotation values

### Import Clinical Data Sequence Diagram

The sequence diagram below shows the sequence for importing the clinical data.

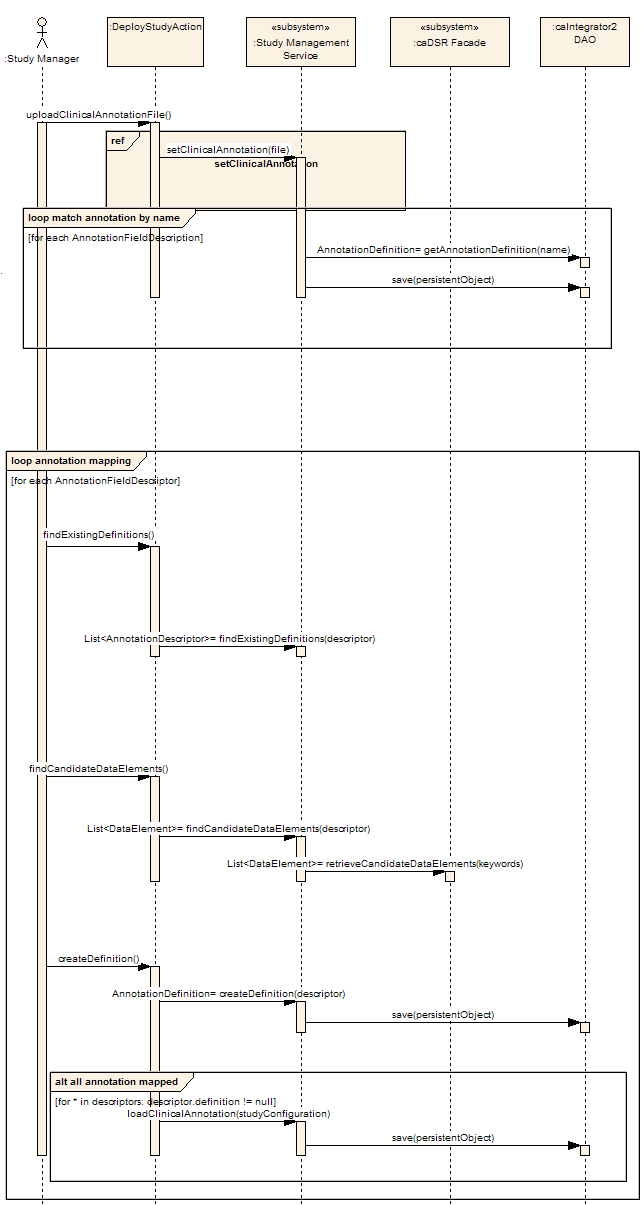


Figure – Study Management Service Import Clinical Data Sequence Diagram

### Configure Genomic Data Process Flow

The study manager configures genomic data for import in the following way:

* The Study manager selects an experiment from caArray to use for this study
* The Study manager selects the samples from that experiment that are relevant for this study
* The sample data is retrieved and associated to the study in caIntegrator2
* The Study manager then associates each sample to a specific subject on the study

### Configure Genomic Data Sequence Diagram

The sequence diagram below shows the sequence for configuring the genomic data.

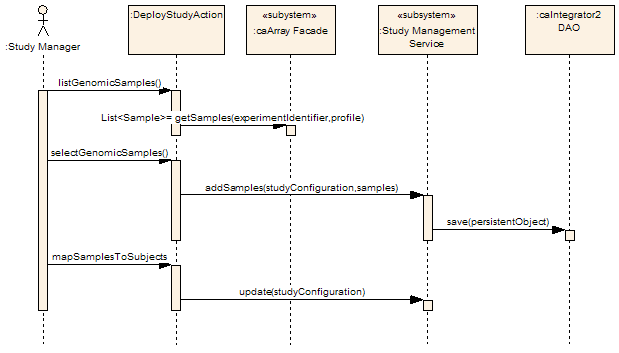


Figure – Study Management Service Configure Genomic Data Sequence Diagram

### Configure Imaging Data Process Flow

The study manager configures imaging data for import in the following way:

* The Study manager selects a trial data Provenance from NCIA to use for this study
* The Study manager selects the image series from that Provenance that are relevant for this study
* The Study manager then associates each image series to a specific subject on the study
* (Optional) The Study manager selects the option to add image annotations to the study
  + The Study manager uploads an image annotations file in CSV format
    - The system validates the file and begins the creation of the first image annotation (loop)
      * The system searches for the annotation in the caIntegrator2 database for reuse
      * (Optional) The system searches for the annotation as a Data Element in caDSR for reuse and stores the caDSR public ID and version
      * (Optional) No matches were found so the Study Manager provides a definition of this annotation and if applicable a list of permissible values
      * The system stores this annotation definition in the database
    - The system then iterates through the image data file and allows the Study manager to associate each value to an image series

### Configure Imaging Data Sequence Diagram

The sequence diagram below shows the sequence for configuring the imaging data.

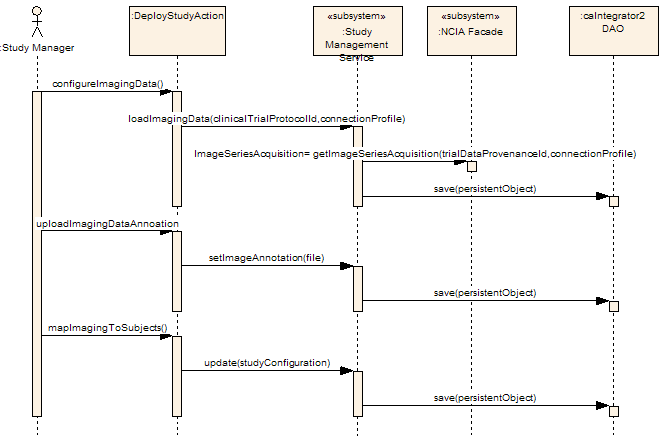


Figure – Study Management Service Configure Imaging Data Sequence Diagram

## Technology Stack

The following technologies or tools are used in the caIntegrator2 application or in the development of the solution:

* JDK 1.5.0
* JBoss 4.0.5.GA
* MySQL 5.0.5
* caGrid 1.1
* Enterprise Architect 6.1+

# Security and Security Management

## Chapter Organization and Content

Topics covered in this chapter include:

* Software Authentication
* Software Authorization

## Software Authentication

The caIntegrator2 application will make use of the Common Security Module (CSM) for performing authentication. All users will be required to login to the application before using it. The basic installation of caIntegrator2 will create an admin account that will allow for the creation of other user accounts. As a function of using CSM all usernames and passwords will be stored in the database. Passwords will be stored encrypted.

## Software Authorization

The caIntegrator2 application will also make use of CSM for performing authorization. We will develop our own user provisioning interface for setting up and defining roles and associating users to roles. When studies are created the study manager will be allowed to specify which data is publicly displayable and what needs to be protected via authorization. The authorization checks will be done within the business logic of the application using the CSM APIs.

# Information Model

## Chapter Organization and Content

Topics covered in this chapter include:

* caIntegrator2 Model Development
* caIntegrator2 Modeling Artifacts
* caIntegrator2 Object Model

## caIntegrator2 Model Development

The initial object model was developed in a brainstorming/modeling session done by the development team given the requirements we knew at the time. The object model for handling the annotations (particularly for the dynamic clinical data) was presented to the Arch/VCDE mentors and discussed at length with them and the caDSR team to ensure that we leveraged as much existing infrastructure and also maintained a semantically comparable data set for later requirements to compare across studies. The entire object model including the annotations was presented and reviewed by SMEs and developers to ensure that everything was properly modeled and nothing left out. The object model was then directly transformed to a data model and mapped using caAdapter and then run through the caCORE SDK API generation process. As the model continues to evolve the mapping and API generation will be maintained.

## caIntegrator2 Modeling Artifacts

The object model and data model were created and are maintained in EA. This model can be found at:

<https://gforge.nci.nih.gov/plugins/scmsvn/viewcvs.php/trunk/docs/analysis_and_design/models/caintegrator2_client_model.EAP?root=caintegrator2&view=log>

## caIntegrator2 Object Model

The following diagrams present the object model broken up into functional groupings.



Figure - caIntegrator2 Annotations Model



Figure - caIntegrator2 Application Model



Figure - caIntegrator2 Genomic Model



Figure - caIntegrator2 Imaging Model



Figure - caIntegrator2 Translational Research Model

# Analysis Service Interface API

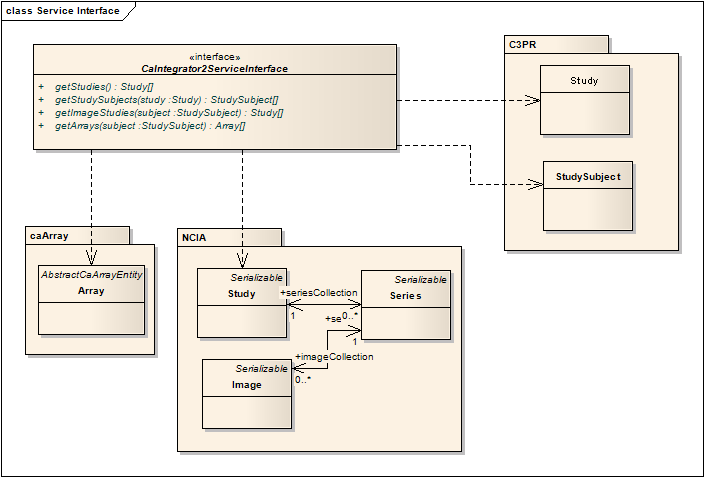
## Chapter Organization and Content

Topics covered in this chapter include:

* caIntegrator2 Service API

## caIntegrator2 Service API

caIntegrator2 will expose a simple analysis grid service API. As an entry point, the service will allow clients to discover the publicly visible studies managed in a caIntegrator2 instance and the subjects associated with those studies. Clients may then use the service API to determine what array data from caArray is associated with a subject as well as imaging data from NCIA. The API has been designed to be as simple and straightforward as possible with the intention of building out further functionality as it is requested by clients. It should be noted that it is expected that there will be very limited use of the service API and that clients will generally wish to use the web UI.



Appendix A References

## Technical Manuals/Articles

1. National Cancer Institute. “caCORE SDK 3.2 Programmers Guide”,

<http://ncicb.nci.nih.gov/infrastructure/cacoresdk>

1. Java Bean Specification:

<http://java.sun.com/products/javabeans/docs/spec.html>

1. Foundations of Object-Relational Mapping:

<http://www.chimu.com/publications/objectRelational/>

1. Object-Relational Mapping articles and products:

<http://www.service-architecture.com/object-relational-mapping/>

1. Hibernate Reference Documentation:

<http://www.hibernate.org/hib_docs/reference/en/html/>

1. Basic O/R Mapping:

<http://www.hibernate.org/hib_docs/reference/en/html/mapping.html>

1. Java Programming:

<http://java.sun.com/learning/new2java/index.html>

1. Javadoc tool:

<http://java.sun.com/j2se/javadoc/>

1. Junit:

<http://junit.sourceforge.net/>

1. Extensible Markup Language:

<http://www.w3.org/TR/REC-xml/>

1. XML Metadata Interchange:

<http://www.omg.org/technology/documents/formal/xmi.htm>

## caBIG Material

1. caBIG:

<http://cabig.nci.nih.gov/>

1. caBIG Compatibility Guidelines:

<http://cabig.nci.nih.gov/guidelines_documentation>

## caCORE Material

1. caCORE:

<http://ncicb.nci.nih.gov/core>

1. caBIO:

<http://ncicb.nci.nih.gov/core/caBIO>

1. caDSR:

<http://ncicb.nci.nih.gov/core/caDSR>

1. EVS:

<http://ncicb.nci.nih.gov/core/EVS>

1. CSM:

<http://ncicb.nci.nih.gov/core/CSM>

Appendix B Glossary

| **Term** | **Definition** |
| --- | --- |
| API | Application Programming Interface |
| caBIG | cancer Biomedical Informatics Grid |
| caBIO | Cancer Bioinformatics Infrastructure Objects |
| caCORE | cancer Common Ontologic Representation Environment |
| cardinality | Cardinality describes the minimum and maximum number of associated objects within a set |
| CDISC | Clinical Data Interchange Standards Consortium |
| CSM | Common Security Module |
| CTMS | Clinical Trial Management Systems |
| CVS | Concurrent Versions System |
| DAO | Data Access Objects |
| DAS | Distributed Annotation System |
| EA | Enterprise Architect |
| EBI | European Bioinformatics Institute |
| EVS | Enterprise Vocabulary Services |
| GULCCC | Georgetown University Lombardi Cancer Center |
| HIPPA | Health Insurance Portability and Accountability Act |
| HTTP | Hypertext Transfer Protocol |
| ISO | International Organization for Standardization |
| JAAS | Java Authentication and Authorization Service |
| JAR | Java Archive |
| Javadoc | Tool for generating API documentation in HTML format from doc comments in source code (<http://java.sun.com/j2se/javadoc/>) |
| JBI | Java Business Integration |
| JDBC | Java Database Connectivity |
| JET | Java Emitter Templates |
| JMI | Java Metadata Interface |
| JSP | JavaServer Pages |
| JUnit | A simple framework to write repeatable tests (<http://junit.sourceforge.net/>) |
| LDAP | Lightweight Directory Access Protocol |
| LLT | Log Locator Tool |
| LOINC | Logical Observation Identifier Names and Codes |
| MedDRA | Medical Dictionary for Regulatory Activities |
| metadata | Definitional data that provides information about or documentation of other data. |
| Multiplicity | Multiplicity of an association end indicates the number of objects of the class on that end may be associated with a single object of the class on the other end |
| NCI | National Cancer Institute |
| NCICB | National Cancer Institute Center for Bioinformatics |
| OMG | Object Management Group |
| ORM | Object Relational Mapping |
| RDBMS | Relational Database Management System |
| SDK | Software Development Kit |
| Semantic connector | A development kit to link model elements to NCICB EVS concepts. |
| SIG | Special Interest Group |
| SQL | Structured Query Language |
| UI | User Interface |
| UID | User Identification |
| UML | Unified Modeling Language |
| UMLS | Unified Medical Language System |
| UPT | User Provisioning Tool |
| URL | Uniform Resource Locators |
| WAR | Web Application Archive |
| WSDL | Web Services Description Language |
| XMI | XML Metadata Interchange (<http://www.omg.org/technology/documents/formal/xmi.htm>) - The main purpose of XMI is to enable easy interchange of metadata between modeling tools (based on the OMG-UML) and metadata repositories (OMG-MOF) in distributed heterogeneous environments |
| XML | Extensible Markup Language (<http://www.w3.org/TR/REC-xml/>) - XML is a subset of Standard Generalized Markup Language (SGML). Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML |

Index

APIs, 10

Architecture, 11

component, 1, 4, 9, 12, 13, 14

Database Model, 25, 30

Features, 9, 10

Grid, 23, 35

Installation, 8

Object Model, 25, 30

Process Flow, 16, 18, 20

Requirements, 5, 8, 9

Security, 5, 24, 35

Sequence Diagram, 16, 18, 20, 21, 22

Use Case, 4, 8, 15

Web Service, 36

WSDL, 36