caCORE RESTFUL WRAPPER

Scope document

Version Number:

Version Date: 1/2/2013

VERSION HISTORY

The most current version of this document is located in the SDK SVN <https://ncisvn.nci.nih.gov/svn/cacoresdk/trunk/projects/docs>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version Number** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Description of Change** |
| 1.0 | Prasad Konka | *10/23/2012* | *<name>* | *<mm/dd/yy>* | *<description of change>* |
| 1.1 | Prasad Konka | 11/02/2012 |  |  | Based on comments from Sichen Liu and Zhengwu Lu |
| 1.2 | Prasad Konka | 12/5/2012 | Robert Shirley | 12/5/2012 | With comments |

TABLE OF CONTENTS

[1 Introduction 4](#_Toc342050276)

[1.1 Purpose of the SCOPE Document 4](#_Toc342050277)

[1.2 caCORE SDK 4](#_Toc342050278)

[2 Scope description 5](#_Toc342050279)

[2.1 RESTFul Wrapper 5](#_Toc342050280)

[1.2 Working with EJB 6](#_Toc342050281)

[3 DELIVERABLES 6](#_Toc342050282)

[3.1 RESTful Wrapper 6](#_Toc342050283)

[4 EXCLUSIONS 6](#_Toc342050284)

[5 CONSTRAINTS 6](#_Toc342050285)

[5.1 User Interface 6](#_Toc342050286)

[5.2 Mapping Generator 7](#_Toc342050287)

[5.3 RESTful Wrapper Generator 7](#_Toc342050288)

[6 ASSUMPTIONS 7](#_Toc342050289)

[7 ACCEPTANCE CRITERIA 7](#_Toc342050290)

[TBDAppendix A: Scope Statement Approval 7](#_Toc342050291)

[APPENDIX B: REFERENCES 9](#_Toc342050292)

# Introduction

## Purpose of the Document

This scope document provides a detailed description of the project, deliverables, constraints, exclusions, assumpitons, and acceptance criteria.

## caCORE SDK

The National Cancer Institute (NCI) Center for Biomedical Informatics and Information Technology (CBIIT) provides biomedical informatics support and integration capabilities to the cancer research community. CBIIT has created the caCORE Software Development Kit or caCORE SDK, a data management framework designed for researchers who need to be able to navigate through a large number of data sources. caCORE SDK is CBIIT’s platform for data management and semantic integration, built using formal techniques from the software engineering and computer science communities.

By providing a common data management framework, caCORE SDK helps streamline the informatics development throughout academic, government and private research labs and clinics. A caCORE SDK generated system is built on the principles of Model Driven Architecture (MDA) and specifically Model Driven Development (MDD) and n-tier architecture and consistent API. The use of MDA and n-tier architecture, both standard software engineering practices, allows for easy access to data, particularly by other applications.

The caCORE SDK generated runtime system's infrastructure exhibits an n-tiered architecture with client interfaces, server components, backend objects, data sources, and additional backend systems. SDK RESTful interface is built to use existing architecture and based on JAX-RS standard to serve client requests over HTTP and return formatted results in XML format. SDK RESTful implementation supports typical CRUD operations:

1. GET: Read access of the resource without side-effects.
2. PUT: create a new resource.
3. DELETE removea a resource.
4. POST update an existing resource.

The major advantages of SDK RESTful implementation are:

1. They are highly reusable across platforms (Java, .NET, PHP, etc) since they rely on basic HTTP protocol
2. They use basic XML instead of the complex SOAP XML and are easily consumable
3. Based on MDA and easy to build fully deployable web application
4. Built-in security with CSM

SDK RESTful implementation requires users to start with a UML model to generate the applicaiton.This step is a potential barrier for many of users in NCIP community to generate a data service and to share data. The communit\y has built data services based on different technologies and the data exchange between the application built using these services is not an easy task. RESTful implementation provides a standardized interface and protocal to service client application across platforms. SDK RESTful Wrapper is aimed to lower the barrier to create RESTful services based on existing data sources. The wrapper will provide interface in RESTful manner while it internally maps with existing data source to query data.

# Scope description

## RESTFul Wrapper

A Wrapper encapsulate legacy logic with high technology coupling requirements, resulting in a creation of implementation coupling throughout all service consumer programs.

In Java world, typically, other than RESTful resources, data is shared through data services built as EJB, SOAP Services or Java API. RESTful wrapper is a way for legacy applications to take advantage of RESTful features to facilitate applications to share their data quickly in a standardized manner. The RESTful Wrapper would encapsulate existing data service types mentioned here to serve client requests in a RESTful manner.

RESTFul Wrapper Generator would let users to define RESTful resources and map them to existing SOAP Web service or EJB. The Wrapper generator then would generate a mapping file that will be used by code generator to generate deployable RESTful resources.

The RESTful Wrapper Generator can be divided into three major parts. Each part is described below.

1. **User Interface**

RESTful wrapper generator should provide easy to use User Interface to define a RESTful resource and map it to existing data service (SOAP or EJB). RESTful resource generation starts with uploading a POJO or JAR of POJOs. Uploaded POJO(s) should be presented in a tree view with Create, Upload, Delete and Read methods as children for each resource. UI should

let user to remove any uploaded POJOs from the tree structure.

1.1 **Working with WSDL**

User Interface should let user to load a WSDL that should open in a new panel in a split window with POJO tree structure on the otherside. RESTful wrapper generator should parse WSDL to present in tree structure with operations supported by given endpoint. User should now be able to drag an operation from WSDL pane to corresponding resource in POJO pane. Mutiple operations can be associated with one RESTful resource operation. Any associations created between RESTful resource and SOAP service operations should be make visible on the UI. The UI should let users to remove any associations created.

## 1.2 Working with EJB

User Interface should let user to upload EJB jar that has Home, Remote, Impl and deployment descriptor. UI should read deployment descriptor and introspect remote interface for EJB operations. These operations should be displayed in tree structure. User should be able map between EJB operation and RESTful resource operation by drag and drop from one end to another. Any associations created between RESTful resource and EJB operations should be make visible on the UI. The UI should let users to remove any associations created.

1. **Mapping Generation**

After mapping is done, user should be able to generate a mapping XML file that should capture all information about resources, operations, SOAP or EJB service details and mappings. This information will be used by RESTful resource generator to generate necessary artifacts. Operations left without any association should be ignored during mapping generation.

1. **RESTful Resource Generation**

RESTful resource generator should validate mapping file for correctness and use it to generate RESTful resources and schema files. It should also generate a project with build script for user to execute and generate a deployable war file.

# DELIVERABLES

## RESTful Wrapper

RESTFul Wrapper is a separate tool to SDK and it should versioned and source controlled in NCI SVN. It should released with instructions and supporting documentation.

# EXCLUSIONS

# CONSTRAINTS

## User Interface

* 1. Editing an existing mapping file through UI is not supported in this release.
  2. UI would only support SOAP and EJB data sources for this release.
  3. UI would not do any validations on correctness or working condition of given SOAP or EJB data services.
  4. UI would not do validation on correctness of information entered into text fields used for mapping.
  5. UI would not support working with EJBs based on v3.0 specification annotations.

## Mapping Generator

### Mapping generator would do syntax validation on mapping XML against XSD

* 1. UI would not do any validations on correctness or working condition of given SOAP or EJB data services in the mapping file.

## RESTful Wrapper Generator

### RESTful Wrapper generator would use avialble open source tools to generate client code for a given SOAP or EJB descriptors. These descriptors should be valid and any errors generated by client code generation tool should be addressed by the user.

# ASSUMPTIONS

# ACCEPTANCE CRITERIA

TBDAppendix A: Approval

The undersigned acknowledge that they have reviewed the SDK 4.5.1 **scope statement** and agree with the information presented within this document. Changes to this scope statement will be coordinated with, and approved by, the undersigned, or their designated representatives.

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: |  |
| Print Name: | Zhengwu Lu |  |  |
| Title: | Technical Project Manager |  |  |
| Role: | Project Manager |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: |  |
| Print Name: | Robert Shirley |  |  |
| Title: | Director |  |  |
| Role: | Project Sponsor |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: |  |
| Print Name: |  |  |  |
| Title: |  |  |  |
| Role: |  |  |  |

APPENDIX B: REFERENCES

[Insert the name, version number, description, and physical location of any documents referenced in this document. Add rows to the table as necessary.]

The following table summarizes the documents referenced in this document.

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Description** | **Location** |
| *<Document Name and Version Number>* | *<Document description>* | *<URL or Network path where document is located>* |
|  |  |  |
|  |  |  |