Requirements Specification

Bioconductor

Fred Hutchinson Cancer Research Center

DOCUMENT REVISION HISTORY

Version Number	Date	Description
1.0	April 2006	• Draft

Background/Summary

This module exposes Bioconductor package functionality as web services available through caGRID. Use cases include Biocondcutor package development as a web service, semantic annotation, deployment management, and web service use.

Table of Content

1 Introduction

1.1 Overview

Bioconductor is a collection of open-source software components based on the R programming language and used for gene expression and other high-throughput analysis in molecular biology. R packages are collections of algorithms grouped to facilitate particular analyses. This module allows R package developers to expose the functionality of their package as web services, specifically available as analytic services on caGRID.

1.2 Related Documents

None.

2 Actors and Goals

2.1 Bioconductor package developer

Develop and expose Bioconductor functionality as a web service.

2.2 Semantic annotation team

Semantically annotate identified Bioconductor package web service functionality to facilitate use as a caGRID analytic service.

2.3 Deployment manager

Deploy semantically annotated Bioconductor packages as caGrid analytic services.

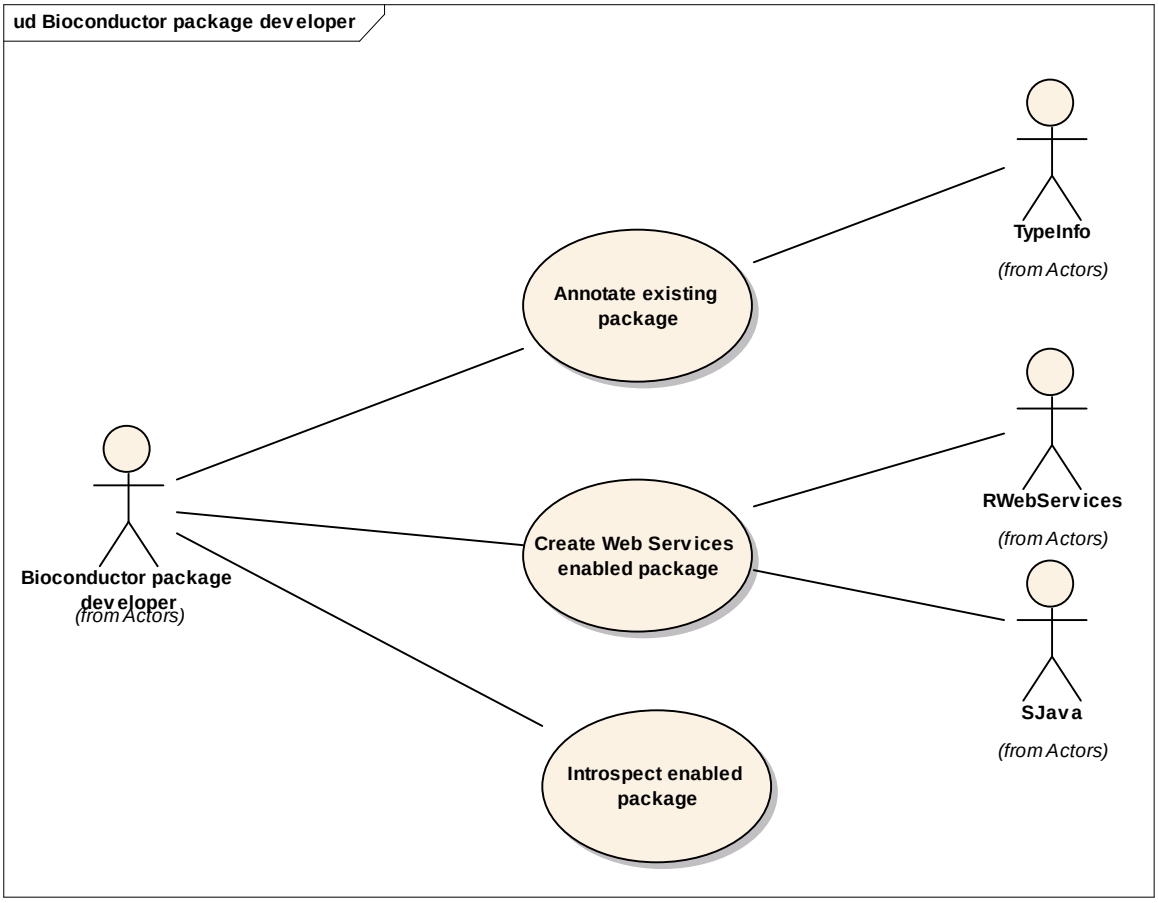
2.4 Web service user

Use Bioconductor analytic service in a caGrid workflow.

3 Use Cases

3.1 Bioconductor package developer

3.1.1 Use Case Model



3.1.2 Brief Description

A Bioconductor package is a group of related functions, documentation, and data that perform specific statistical analyses. Bioconductor packages are usually authored by one or a few individuals, and address specialized or cutting-edge needs in statistical analysis.

This use case involves a Bioconductor package developer who desires to make all or a portion of the functionality of their package available as a web service within the caGrid environment.

3.1.3 Primary Actor

The primary actor in this use case is a computationally- and statistically capable individual wishing to produce or modify an R package to be deployed as a web service in caGRID. The individual is familiar with R, but has no special skills related to web services.

3.1.4 Preconditions

3.1.4.1 Functioning R system

The R package developer requires a functioning R system. This includes a development tool chain described in detail in documentation associated with the R project.

3.1.4.2 Primary Bioconductor package

The Bioconductor package developer has authored a package to perform statistical analysis. The Bioconductor package will contain R codeand documentation, and may contain C or Fortran code. Prior to adaptation to the web services environment, the package satisfies the normal conditions placed on R packages, compiling, linking, and installing into the developer's standard R system without errors or warnings.

3.1.4.3 Additional Bioconductor packages

TypeInfo, a Bioconductor package to provide strongly typed access to R functions and methods.

RWebServices, a Bioconductor package to convert other R packages to web services. RWebServices translates R objects, methods, and functions to corresponding Java classes and methods.

SJava, a Bioconductor and Java package that provides infrastructure for the R/Java interface.

3.1.5 Basic Flow of Events

The R package developer initiates flow of events by identifying those aspects of their package that are most suitable for use in a web services context.

The Bioconductor package developer uses TypeInfo to apply type specification to the functions to be web-service enabled. This step is required because R is not a strongly typed language. Type specification is conducted in the R program, and integrated into R packages by including appropriate statements in the (R) source

code for the package. The steps required for this are very familiar to the R developer.

The developer initiates the next step in the workflow by providing a list of TypeInfo-enabled functions to the RWebServices package. RWebServices uses type specification in conjunction with method and function definitions contained in the TypeInfo-enabled package (and other packages the developer may have required) to generate a collection of Java classes corresponding to objects in the TypeInfo-enabled package. The Java classes can integrate into SJava for evaluation.

The developer then invokes Java introspection tools on the classes constructed by RWebServices to construct UML domain models expressed as XMI.

3.1.6 Postconditions

3.1.6.1 Web services interface to Bioconductor package

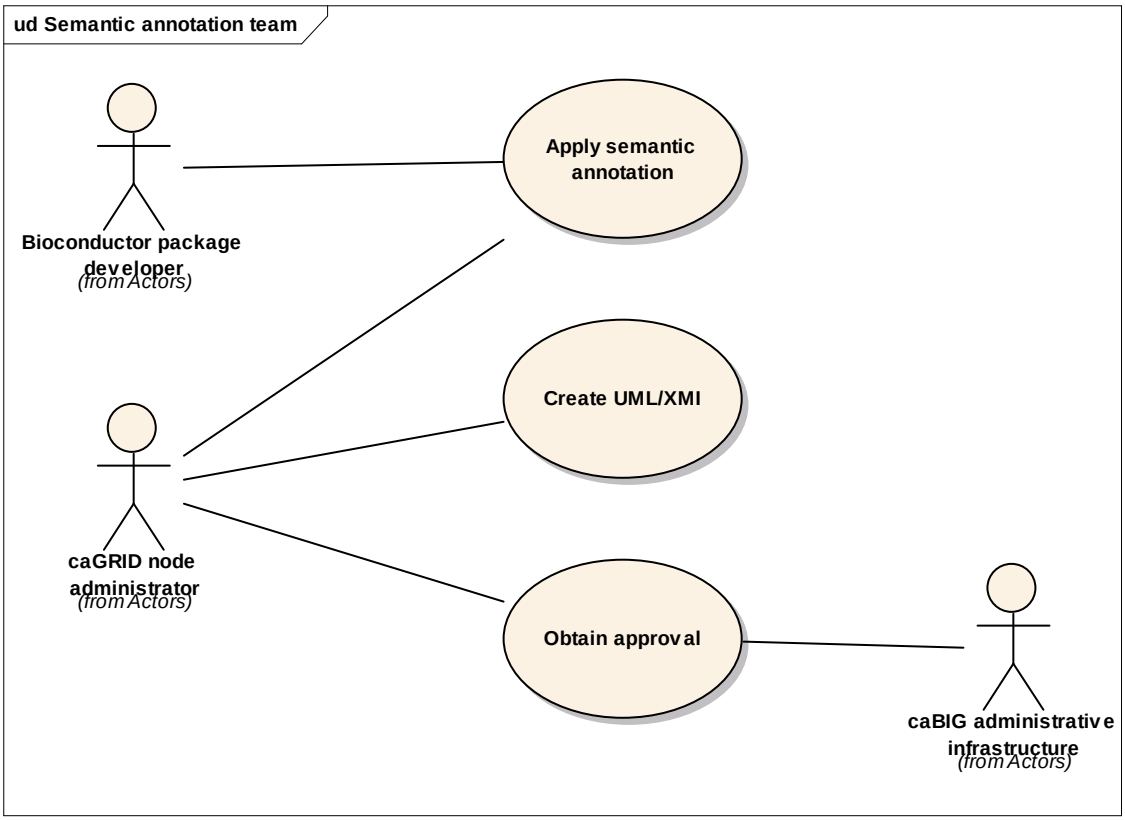
The postcondition of this use case is a Java interface to specific functionality of a Bioconductor package, coupled with UML domain models expressed as XMI.

3.1.7 Special Requirements

None.

3.2 Semantic annotation team

3.2.1 Use Case Model



3.2.2 Brief Description

This use case adds semantic annotation to a web services enabled Bioconductor package, so that the package can be incorporated into caGrid. This use case is a collaboration between the Biocondcutor package developer (an expert in a narrowly defined statistical analysis) and a caGRID node administrator (familiar with semantic concepts important in caGRID). The semantic annotation team collaborates to associate argument and return types with terms from semantically controlled vocabularies.

3.2.3 Primary Actor

The primary actor is the individual responsible for exposing Bioconductor functionality on caGrid. This may be the package developer, or the caGrid node administrator.

3.2.4 Secondary Actors

Either the package developer or caGrid node administrator is a secondary actor, depending on the primary actor.

3.2.5 Preconditions

3.2.5.1 RWebServices enabled package

The RWebServices enabled package is the postcondition of use case 3.1.

3.2.5.2 Semantic annotation tools in caCORE SDK

3.2.6 Basic Flow of Events

The primary actor uses caCORE SDK tools and procedures to apply semantic annotation to the UML domain models. This step uses the caCORE Enterprise Vocabulary Services (EVS) APIs hosted at the NCI to search the NCI Thesaurus for the appropriate concepts.

The semantically annotated UML domain model is exported as XMI.

The primary actor gains approval of semantically annotated UML domain model and XMI from the caBIG infrastructure.

3.2.7 Postconditions

3.2.7.1 Semantically annotated RWebServices package

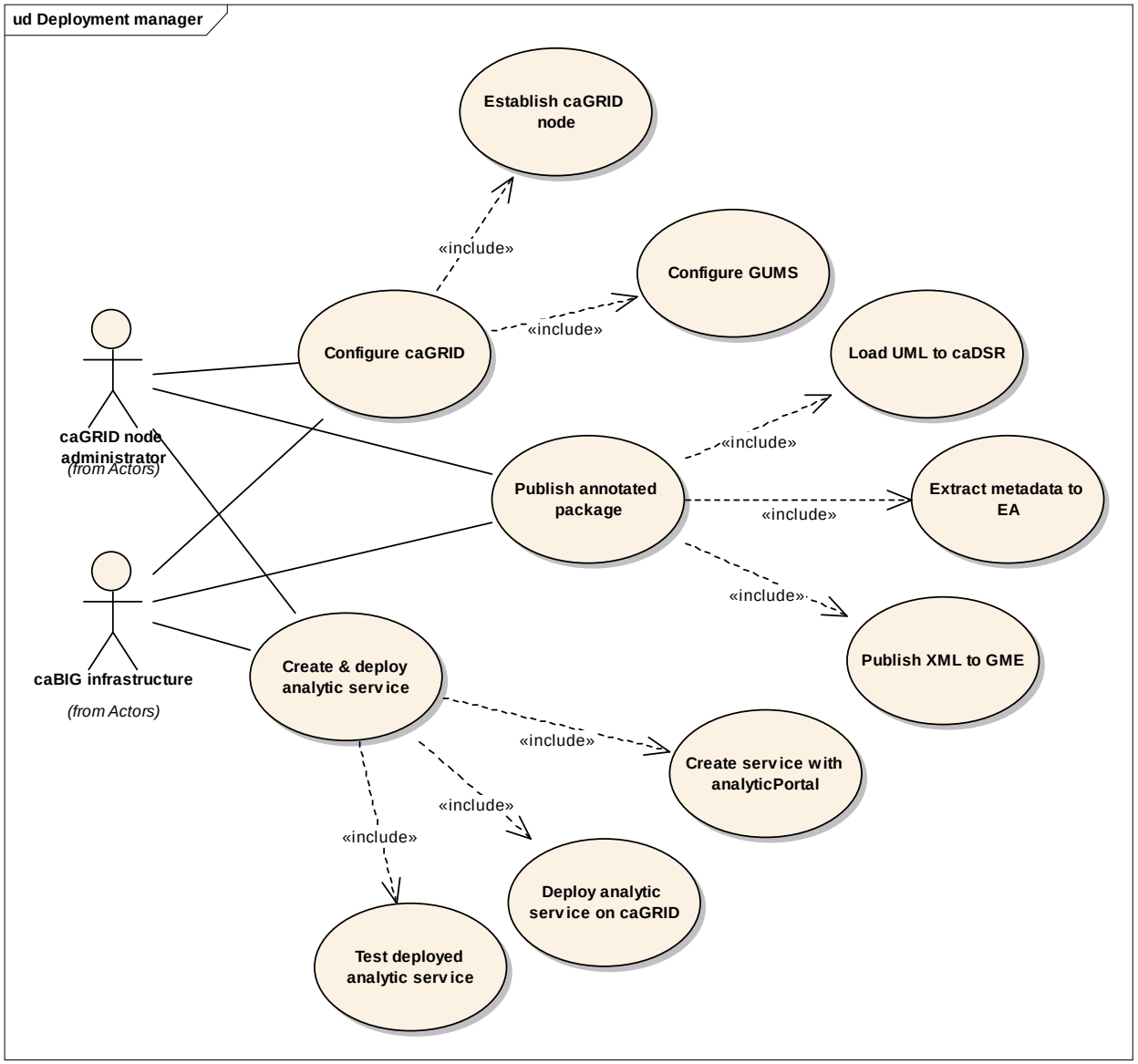
The postcondition consists of the Bioconductor package(s), RWebServices Java wrappers, and a semantically annotated and approved UML domain model expressed as XMI.

3.2.8 Special Requirements

None.

3.3 Deployment manager

3.3.1 Use Case Model



3.3.2 Brief Description

The deployment manger takes a set of java programs generated from RWebServices as input, and generates a valid caGRID analytical service.

3.3.3 Primary Actor

Deployment manager.

3.3.4 Preconditions

3.3.4.1 Semantically annotated RWebServices enabled package

This is the postcondtion of use case 3.2.

3.3.4.2 Established caGRID node

This use case presumes that a caGRID node is already established. The node likely includes existing data and analytic services.

3.3.4.3 caGRID deployment tools and facilities

Tools and facilities include Cancer Data Standards Repository (caDSR), UML Domain Model Loader (part of caCORE SDK), the Grid User Management Service (GUMS), Global Model Exchange (GME) and the GME Viewer GUI, analyticPortal, ant script for analytic service deployment (from caGRID).

3.3.5 Basic Flow of Events

The deployment manager configures GUMS, running either locally or with a local proxy to the GUMS service of NCI.

The deployment manager transforms and loads the semantically annotated UML domain models into caDSR. This step is facilitated by the UML Domain Model Loader.

The metadata resulting from the previous step is extracted to an XML schema using Enterprise Architect.

The deployment manager publishes the XML Schema into the GME using GME Viewer. The first step is to connect GME Viewer to a GME Service (at NCI or elsewhere). The second step is to publish the XML schemas to the GME service. This inserts the XML schemas into the XML database on GME server.

The deployment manager creates an analytic service using analyticPortal. The first step is to create an analytical service. The second step is to add methods to the analytical service (this can be done using a GUI, or more effectively through the command line). A method description includes the method name, name and type of input parameters and return data. All data types are selected from published XML Schemas on GME servers. analyticalPortal automatically generates Java skeletons for the methods, data types, and web service functionality. The final step is to insert code into the skeletons to provide implementation.

The deployment manager deploys the analytic service on a running web server. This is done with an ant script available in caGRID, and involves copying appropriate portions of the analytic services directory to certain server directories.

The deployment manager validates the analytic service using an established test suite and protocol.

3.3.6 Postconditions

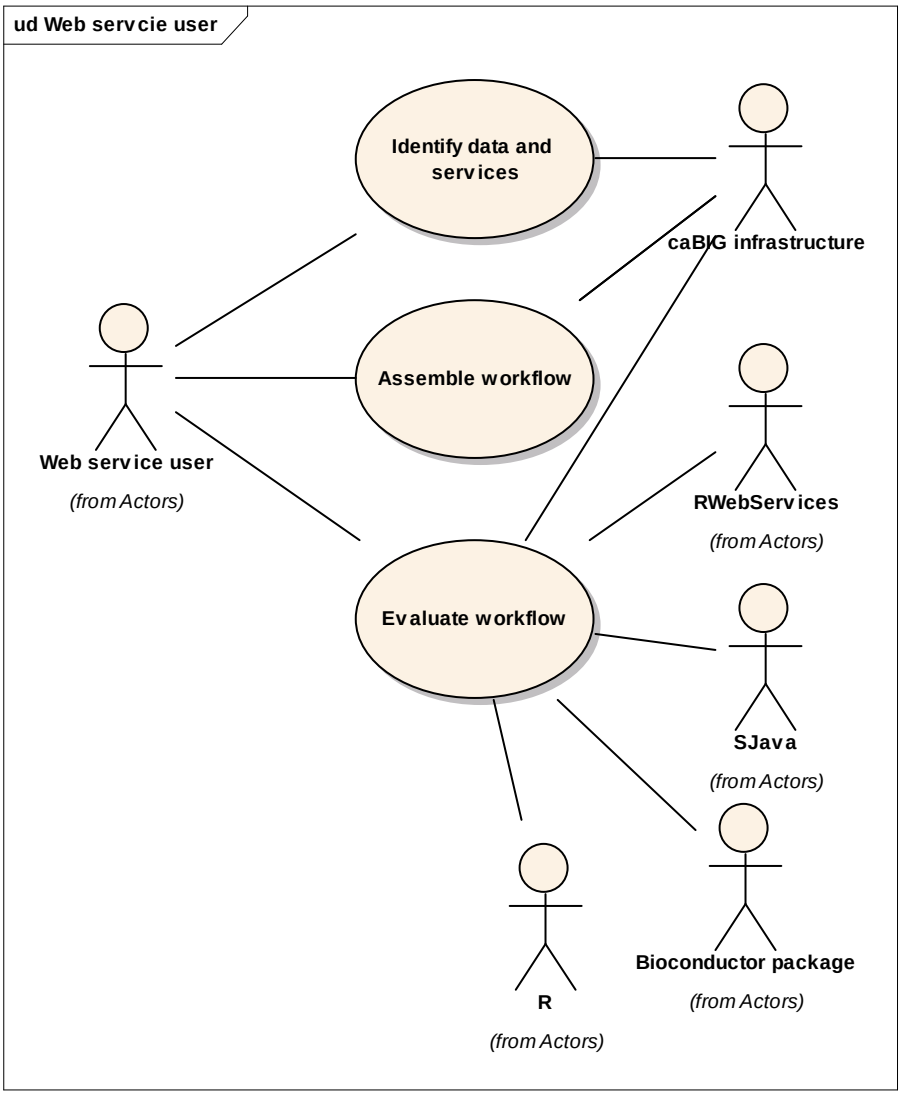
3.3.6.1 Biocondcutor package functionality available as caGRID analytic service

3.3.7 Special Requirements

None.

3.4 Web service user

3.4.1 Use Case Model



3.4.2 Brief Description

This use case involves caGRID-deployed RWebServices packages as analytic services in a caGRID workflow.

IMPORTANT NOTE: this use case represents a key component of the project, but the workflow and client tools (‘caBIG infrastructure) are NOT part of the deliverables for this project.

3.4.3 Primary Actor

The web service user is a clinician, statistician, or other practitioner with access to caGRID workflow tools, data, and analytic services. The user may have no specialized programming or systems administration knowledge.

3.4.4 Secondary Actors

caGRID or local data source.

caGRID or other workflow client.

3.4.5 Preconditions

3.4.5.1 caGRID data source and workflow tools

3.4.5.2 caGRID-deployed RWebServices package

3.4.6 Basic Flow of Events

The web service user identifies data and analytic services to be used.

The web service user assembles a caGRID workflow that describes how data and analytic services are combined.

The web service user executes the caGRID workflow.

RWebServices participate in the workflow by accepting semantically correct arguments and returning semantically correct values.

Workflow can involve parallel execution of tasks. The caGRID workflow coordinates division of tasks between available compute resources.

Workflow can involve extended computation. The caGRID workflow provides mechanisms for session management, including retrieving workflow outcomes.

3.4.7 Postconditions

3.4.7.1 Semantically correct results

The usual postcondition of each step involving RWebServices will be a single semantically correct object, representing the result of the service.

3.4.7.2 Exceptions

Errors during RWebServices evaluation, due to malformed data, unanticipated computational problems, or limitations of the RWebServices algorithm, result in exceptions containing descriptive information about the nature of the difficulty.

3.4.7.3 Side-effects

RWebServices may generate side-effects. These include informational messages (R 'warning' and 'message') about special conditions encountered during evaluation, and secondary output (e.g., diagnostic graphical plots) produced during function evaluation. These are reported as WSDL attachments, and are made available to the web services user through the caGRID workflow.

3.4.8 Special Requirements

3.4.8.1 caGRID workflow client

The forgoing presumes a caGRID workflow client, with specific abilities that include division of tasks between available computational resources, session management, handling exception, and reporting side effects. No such caGRID workflow client is currently available.

4 Overall Requirements

4.1 Requirements Prioritization

Priority	Use Case Name	Supp Requirement Ref #
1	Bioconductor package developer	
3	Semantic annotation team	
2	Deployment manager	
4	Web service user	

5 Data Definitions

Bioconductor package – a valid collection of statistical and other analytic tools written in the R programming language.

TypeInfo, SJava –existing Bioconductor packages providing infrastructure for this project.

6 Sign Off

6.1 Approval

Workspace General Contractor Rep	*Print Name*	*Date*
Architecture Workspace Rep	*Print Name*	*Date*
VCDE Workspace Rep	*Print Name*	*Date*
Workspace Working Group Rep	*Print Name*	*Date*
NCICB Rep	*Print Name*	*Date*