This page last changed on Sep 30, 2008 by scamp.

JIRA Services Specification

1. Scope

1.1 Overview

This component provides a wrapper to the JIRA Management component, exposing the main management functionality as webservices. This component must also provide client classes that will allow for easy remote access to the service methods.

1.2 Logic Requirements

1.2.1 Interfaces

The interfaces for this component can be found in the provided tcuml file, in the "Web Services" class diagram. Both Local and Remote EJB3 interfaces are required.

1.2.2 EJB Security

No explicit security is needed for the webservice, as it will be called anonymously, but authentication will happen through the login and logout methods of the management interface the service wraps.

1.2.3 Logging

Each webservice method called should be logged at the DEBUG level, including the parameter information. If errors occur in any call, a message should be logged at the WARNING level before the exception is rethrown.

1.2.4 Webservice Client

This component must define a client class that mimics the API of the service, to make it easy for a user to access the webservices remotely.

1.2.5 Configuration

Configuration must occur through the Configuration API and the Configuration API Object Factory Plugin. For the implementation, the configuration will be read from a file using the @Resource attribute values and the Configuration Persistence component.

1.2.6 Connection

This service needs to fail gracefully if it is unable to connect to JIRA or is otherwise unable to communication with JIRA. The manager will throw a JiraConnectionException if the connection fails. If the connection to JIRA fails, it should continue functioning as a service even while JIRA is unavailable, returning a JIRA communications error' web-service fault back to the calling application.

1.3 Required Algorithms

None

1.4 Example of the Software Usage

This component will be used to access the JIRA Management component remotely.

1.5 Future Component Direction

If the JIRA Management component is updated, this component will be updated to reflect the changes.

2. Interface Requirements

2.1 Software Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

Designs must adhere to the interface diagram definition found in the architecture TCUML file provided [JIRA_Services_Architecture.tcuml]. Changes to the interfaces should be approved in the forum.

2.1.3 Environment Requirements

- Development language: Java1.5
- Compile target: Java1.5 and Java1.6

2.1.4 Package Structure

com.topcoder.jira.webservice
com.topcoder.jira.webservice.bean
com.topcoder.jira.webservice.client

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- The manager interface instances to use

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

None

3.2.2 TopCoder Software Component Dependencies:

- JIRA Management 1.0
- Base Exception 2.0
- Logging Wrapper 2.0

- Object Factory 2.1
- Configuration API 1.0
- Configuration Persistence 1.0
- Object Factory Configuration API Plugin 1.0

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

3.2.3 Third Party Component, Library, or Product Dependencies:

None.

3.2.4 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000
- Windows 2003
- Informix 10.0

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines.

3.4 Required Documentation

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the 'Documentation' tab of the TopCoder UML Tool.