Java JNDI Context Utility 2.0 Requirements Specification

1. Scope

1.1 Overview

Java Naming and Directory Interface (JNDI) provides a common interface to both naming and directory services. The JNDI Context Utility simplifies access to JNDI Contexts, including creating Contexts, manipulating JNDI names and retrieving database connections or JMS resources. Additionally, the component provides a command line and programmatic interface for dumping the JNDI tree to an XML file.

1.2 Logic Requirements

1.2.1 Maintain version 1.0 API

The component will retain the version 1.0 API without external changes (although internal refactoring is permitted).

1.2.2 Instance methods

The new version must allow for creating JNDIUtils instances. Each instance must support the same methods as the current JNDIUtils. However, all methods will be instance methods as opposed to the current static methods.

1.2.3 Configuration

Support for configuring a JNDIUtils instance must come in two implementations.

1.2.3.1 Configuration Manger Implementation

The JNDIUtils should read its configuration from a specified namespace. The configuration should be the same as the current configuration used for the static methods (more options may be added at the designer's discretion), except that the namespace the configuration appears in will not be fixed.

1.2.3.2 Custom xml format

The second configuration implementation for a JNDIUtils instance must read from a custom xml format. The designer will provide an XSD for the format. It should be possible to load a JNDIUtils instance passing an XML document in this format either as an InputStream or as a File.

1.3 Required Algorithms

None Specified

1.4 Example of the Software Usage

An example usage of this component is a website that needs to utilize EJBs which exist on numerous servers. One server is used for user information and another for transaction information. Using the JNDI Context Utility abstracts the context from the developers and makes the details configurable.

1.5 Future Component Direction

None.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None Specified

2.1.2 External Interfaces

None Specified

2.1.3 Environment Requirements

- Development language: Java 1.4

2.1.4 Namespace

com.topcoder.naming.jndiutility

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- None.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

List of JNDI context sources.

3.2.2 TopCoder Software Component Dependencies:

Configuration Manager

**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

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines. Modifications to these guidelines for this component should be detailed below.

3.4 Required Documentation

3.4.1 Design Documentation

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

3.4.2 Help / User Documentation

Relevant design documentation must provide sufficient information regarding component design and usage.