1. Scope

1.1 Overview

This component provides a framework for managing TopCoder clients, projects, and companies. This component provides simple API calls to manipulate and retrieve clients, projects, statuses, and companies, using the Client and Project Entities and DAO component to provide the entities and persistence implementation.

1.1.1 Version

1.0

1.2 Logic Requirements

This component must implement all the functionality shown on the "Manager Class Diagram" in the provided interface TCUML.

1.2.1 Interfaces

The following interfaces will be provided by this component:

- ClientManager
  - This interface defines functionality for managing client information, including CRUD functionality for clients and statuses, searching for clients by name and a provided search filter, and getting clients by status, as well as directly updating a client's status and code name.
- ProjectManager
  - This interface defines functionality for managing project information, including CRUD functionality for projects and statuses, searching for projects by name and a provided search filter, and getting projects by status, as well as getting all projects for a provided client
- CompanyManager
  - This interface defines functionality for managing company information, including CRUD functionality for companies, as well as retrieving clients and projects for a provided company.

1.2.2 Implementations

Implementations of the three interfaces will be provided that leverage the DAO interfaces from the Client and Project Entities and DAO component. The DAO implementations can be created through configuration, but if the default constructors are used to create the manager implementations, the EJB3 DAO implementations should be used as the default.

1.2.2 Logging

Each manager 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 re-thrown.

1.2.2 Configuration

Configuration must occur through the Configuration API and the Configuration API Object Factory Plugin. Besides the normal ConfigurationObject constructors for the classes that involve configuration, constructors should be added that use the Configuration Persistence component to retrieve configuration information saved in configuration files.

1.2.2 ID Generation

When entities are created, the ID Generator must be used to generate unique ID's for each entity being added to the database.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

This component will be used as a direct API for manipulating client and project information.

1.5 Future Component Direction

Further manager implementations can be added.

2. Interface 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. Designers can choose to add more methods to the interfaces, but must keep the ones defined on the diagram as a minimum. Changes to the interfaces should be approved in the forum.

2.1.3 Environment Requirements

- Development language: Java 1.5
- Compile target: Java 1.5 and Java 1.6

2.1.4 Package Structure

com.topcoder.clients.manager
com.topcoder.clients.manager.dao

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- The log to use
- The DAO implementations 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:

- Configuration API 1.0
- Configuration Persistence 1.0.1
- Base Exception 2.0
- Logging Wrapper
- Search Builder 1.4
- ID Generator 3.0
- Client and Project Entities and DAO 1.0
- Object Factory 2.1
- 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

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

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