[TopCoder]

Client Association Services 1.0 Requirements Specification

1. Scope

1.1 Overview

This component will provide services for the following associations:

- Clients and components: meaning that a component (or application) is developed for the client.

- Clients and users: meaning that a user is authorized to work with the client components.

The services provided by this component will enable the creation, deletion and query of those associations, using database persistence.

1.2 Logic Requirements

1.2.1 Operations

The following operations must be provided:

1.2.1.1 Assign Component to Client

A new association will be created between a component and a client

1.2.1.2 Assign User to Client

A new association will be created between a user and a client. The user can be a regular client user or an administrative user for that client.

1.2.1.3 Remove Component from Client

An association between a component and a client will be erased.

1.2.1.4 Remove User from Client

An association between a component and a user will be erased.

1.2.1.5 Get Components for Client

It will return a list of the component ids associated with a client

1.2.1.6 Get Users for Client

It will return a list of the user ids associated with a client

1.2.1.7 Get Role

Given a user id and client id, this operation will retrieve whether the user is a regular user for the client or an admin user.

1.2.1.8 Get Clients for Component

It will return a list of client ids such that those clients are associated with the specified component id.

1.2.1.9 Get Clients for User

It will return a list of client ids such that those clients are associated with the specified user id.

1.2.1.10 Get Components for User

Given a user id, it will look for all the associated clients, and for each of those clients, it will find its components, returning a list of component ids. This should be optimized in order to avoid doing many queries.

1.2.2 Persistence

The component must provide persistence using Hibernate.

The database schema is provided in the appendix.

1.2.3 Session Bean

This component must implement a stateless session bean (with both local and remote interfaces) providing all the operations described in 1.2.1.

Each operation must run with REQUIRED transaction attribute.

1.3 Required Algorithms

None.

1.4 Future Component Direction

None.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None

2.1.2 External Interfaces

None

2.1.3 Environment Requirements

- Development language: Java 5.0
- Compile target: Java 5.0

2.1.4 Package Structure

com.topcoder.controlpanel.clientassociations

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?

- Hibernate
- EJB 3.0

3.2.2 TopCoder Software Component Dependencies:

Auditor 2.0

**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:

Informix Database

3.2.4 QA Environment:

- RedHat Enterprise Linux 4
- JBoss 4.2 GA with EJB 3.0
- Java 1.5
- Informix 10.00.UC 5

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 Poseidon.

4. Appendix

4.1 Database DDL

```
-- client table actually contains more fields, but for the purpose of this components, this is enough.
create table client (
    client_id INT not null,
    name VARCHAR(64) not null
);

alter table client add constraint primary key
            (client_id)
            constraint client_pk;


create table user_client (
    user_id DECIMAL(10,0) not null,
    client_id INT not null,
    admin_ind DECIMAL(1,0)
);

alter table user_client add constraint primary key
    (user_id, client_id)
    constraint user_client_pk;

alter table user_client add constraint foreign key
    (client_id)
    references client (client_id)
    constraint user_client_client_fk;


create table comp_client (
    component_id DECIMAL(12,0) not null,
    client_id INT not null
);


alter table comp_client add constraint primary key
    (component_id, client_id)
    constraint comp_client_pk;

alter table comp_client add constraint foreign key
    (client_id)
    references client (client_id)
    constraint comp_client_client_fk;
```