Requirements Specification

1. Scope

1.1 Overview

The Time Tracker User custom component is part of the Time Tracker application. It provides an abstraction of a user in the system which will be imported from a variety of sources. This component handles the persistence and other business logic required by the application.

1.2 Logic Requirements

1.2.1 Time Tracker Users

All users will be imported from a variety of sources called User Stores. Within the scope of the Time Tracker application, an internal ID will be generated and the original user name will be replicated locally for performance.

Hence, the following user attributes will be maintained:

- User ID – internal Time Tracker ID, also used as primary key for the database
- Name – the original user name from the user store

The basic database schema to store users is as follows:

```
CREATE TABLE Users (
    UsersID          integer NOT NULL,
    Name             varchar(64) NOT NULL,
    CreationDate     datetime NOT NULL,
    CreationUser     varchar(64) NOT NULL,
    ModificationDate datetime NOT NULL,
    ModificationUser varchar(64) NOT NULL,
    PRIMARY KEY (UsersID)
);
```

Note: Depending on the design of user stores, new columns may be added at the designer's discretion.

1.2.2 User Stores

1.2.2.1 Overview

A user store is a kind of storage mechanism, such as LDAP, Active Directory, and databases. They contain an assortment of user profile details. The creation and maintenance of these users are outside the scope of the Time Tracker application. The application will assume these users already exist in some kind of user store and import them into the system.

The design will provide a pluggable framework for the purpose of importing users from different user stores into the Time Tracker application. Also, the relationship between the imported users and the original user store must be maintained.

1.2.2.2 API Requirements

The Time Tracker application will present a generic user interface for importing users. To facilitate this, the pluggable user stores should provide the following operations:

- Connection – a way to connect to the user store
- Enumeration – a way to enumerate all existing users in the user store. Note that this calls for the ability to query, at a minimum, the user names.
- Search – a way to search for specific users by name in the user store
- Authentication – a way to authenticate a user against a given password

Note: Other operations may be added at the designer's discretion.

1.2.2.3 Default User Store

For clients that do not have an existing user store, a default Informix database implementation will be provided. Clients will maintain the database externally. The Time Tracker application will be able to import users from the default implementation just like any other user stores.

The database schema for the default implementation is as follows:

```
CREATE TABLE DefaultUsers (
    DefaultUsersID      integer NOT NULL,
    UserName            varchar(64) NOT NULL,
    Password            varchar(64) NOT NULL,
    FirstName           varchar(64),
    LastName            varchar(64),
    Phone               varchar(20),
    Email               varchar(64),
    CreationDate        datetime NOT NULL,
    CreationUser        varchar(64) NOT NULL,
    ModificationDate    datetime NOT NULL,
    ModificationUser    varchar(64) NOT NULL,
    PRIMARY KEY (DefaultUsersID)
);
```

1.2.3 User Roles

1.2.3.1 Overview

Each user has one assigned role. The design will make use of the Authorization component to support user roles and permissions. The Time Tracker application will make checks when performing operations on behalf of a user.

The following user roles have been identified:

- Super Administrator – this user can perform all functions and view all data
- Human Resource – this user is from the Human Resource Department
- Account Manager – this user manages multiple clients
- Project Manager – this user manages multiple projects
- Internal Employee – this user is a regular employee
- Contractor – this user works on a per contract basis

Note: New roles may be added or modified as per application requirements.

1.2.3.2 API Requirements

The following operations for user roles have been identified:

- Set the role for a user
- Query the role for a user
- Enumerate all available user roles

1.2.4 Pluggable Persistence

All entities defined above will be backed by a database. The design will provide the necessary API to store and retrieve data from the database.

For the initial version, the Informix database system will be used as persistence storage for this component and the Time Tracker application. Other database systems should be pluggable into the framework.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

The Time Tracker application will use this component to perform operations related to user authentication and authorization.

1.5 Future Component Direction

Other database systems maybe plugged in for some client environments. Multiple user stores may be used for the same client environment.

2. Interface Requirements

2.1.1 Graphical User Interface Requirement

None.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java 1.4
- Compile target: Java 1.3, Java 1.4

2.1.4 Package Structure

com.topcoder.timetracker.user

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?

None.

3.2.2 TopCoder Software Component Dependencies:

- Configuration Manager
- DB Connection Factory
- Authentication Factory
- Authorization

**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:

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

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.