Requirements Specification

1. Scope

1.1 Overview

The Document Indexer Persistence component implements the persistence layer as required by the Document Indexer component. The pluggable framework allows different persistence mechanisms to be used. For the initial version, two mechanisms (XML and database) are provided.

This component will implement interfaces defined by the Document Indexer component to persist the necessary document index data. Designers should familiarize themselves with the overall design and API of that component.

1.2 Logic Requirements

1.2.1 Overview

This component implements the persistence API as defined by the Document Indexer component. In particular, the `IDocumentIndexSource` interface is to be followed. Refer to the documentation of that component for more details.

1.2.2 XML Persistence

The design will provide a persistence implementation using XML files. The designer will determine the appropriate XML format to achieve this goal. The corresponding XSD will be provided for validation purpose.

Note: It may be necessary to implement a multi-file strategy to overcome limitations in file sizes.

1.2.3 Database Persistence

The design will provide a persistence implementation using databases. The necessary database schema used to store the information will also be provided. For the initial version, the Informix database system will be used.

Note: Both Data Definition Language (DDL) script and Entity Relationship Diagram (ERD) will be provided to describe the database schema.

1.2.4 Globalization Support

The design should support documents written in multiple languages. The default supported language is English.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

TopCoder has a large collection of text data such as problem statements from previous SRM's. The ability to quickly search through the problem archive for particular words or phrases will be a valuable feature of the web site.

1.5 Future Component Direction

In the future, different persistence implementations will be added.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

Implement the persistence API as defined by the Document Indexer component.

2.1.3 Environment Requirements

- Development language: C# 1.1
- Compile target: C# 1.1

2.1.4 Package Structure

TopCoder.Document.Index.Persistence

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

Database connection should be configurable.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

- .NET Framework 1.1

3.2.2 TopCoder Software Component Dependencies:

- Configuration Manager
- Connection Factory
- Document Indexer

**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:

- Windows 2000
- Windows Server 2003

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.