Logging Wrapper 1.3 Requirements Specification

1. Scope

1.1 Overview

The Logging Wrapper component provides a standard logging API with a pluggable back-end logging implementation. Utilization of the Logging Wrapper insures that components are not tied to a specific logging solution. More importantly, a change to the back-end logging solution does not require a code change to existing, tested components. Support exists for log4j and java1.4 Logger as back-end logging implementations.

The Logging Wrapper 1.3 project will enhance certain areas of the component.

1.2 Logic Requirements

1.2.1 Compatibility with Version 1.2

The new design will enhance the existing Version 1.2 design and maintain the same API. New classes and methods will be added to meet the new requirements.

1.2.2 Composite Logger

The design will provide a new composite logger, which aggregates two or more logger implementations. A call to the `log(…)` method will deliver the log message to all the loggers.

The following operations are required:

- Add a target logger
- Remove a target logger
- Enumerate all target loggers

1.2.3 Database Logger

The design will provide a new logger implementation using a database back-end to persist the log messages. The logger will optionally record the datetime when the log entry is stored. The appropriate schema will be supplied by the designer.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

The initial use of the Logging Wrapper component will be within other TopCoder Software components. This will allow TopCoder Software components to be plugged into an existing environment without requiring the additional configuration and implementation of a specific logging solution.

1.5 Future Component Direction

Additional logger implementations may be added in the future.

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 1.4, Java 1.5

2.1.4 Package Structure

com.topcoder.util.log

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- Database connection information will be configurable

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

**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

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.