Auto Screening Management Requirements Specification

1. Scope

1.1 Overview

Automated screening is a tool that screens submissions automatically. Submissions are put in a queue and will be picked up by standalone screening tools that could potentially run from multiple servers. Screening rules are configurable per project category. Once the submission is screened, the results will be logged.

This component provides the management API to initiate screening task and to query screening results. Another component will be developed as the screening tool.

1.2 Logic Requirements

1.2.1 Screening Task

There should be a way to initiate a screening task. The screening task will comprise of a numeric identifier which can then be used to retrieve the uploaded submission and related information. Each screening task will be associated with a status. Valid statuses for this release includes Pending, Screening, Failed, Passed, Passed with Warning. The task should be set to pending once it's initiated. The ID should be generated with the ID Generator.

Screening task must also include auditing fields of creation/modification operator and timestamp. These fields will not be provided by component users.

1.2.2 Screening Response

Once a screening task is performed, a set of screening response will be produced by the automated screener. The screening response will comprise of response code, fixed response text and dynamic response text. In addition, each response will be associated with a severity code.

1.2.3 Query Screening Tasks

User can query screening tasks for an array of upload identifiers. Screening details will not be retrieved in this mode.

1.2.4 Query Screening Details

User can query screening details for a single upload identifier.

1.2.5 Persistence

The persistence will be an Informix database. Pluggability is not required. The SQL scripts will be provided.

1.3 Required Algorithms

No specific algorithms are required.

1.4 Example of the Software Usage

The component will be integrated into the Online Review application to screen submissions after they are uploaded. Screening will be kicked off immediately after the upload. Results can be retrieved with the API.

1.5 Future Component Direction

None.

2. Interface Requirements

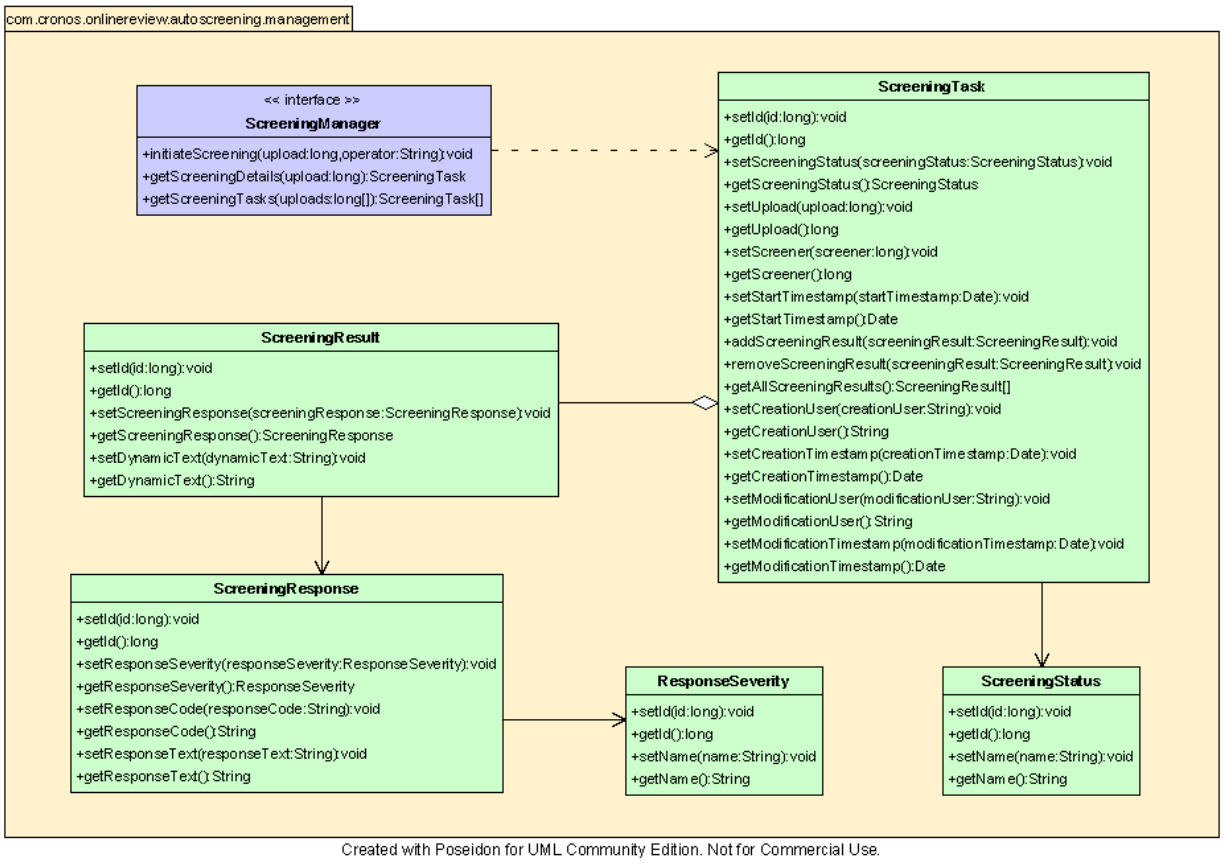
2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

Design must adhere to the interface diagram definition. Designer can choose to add more methods to the classes/interfaces, but must keep the ones defined on the diagram as a minimum. Source files can be found in the distribution.

Auto Screening Management Interface Diagram



Created with Poseidon for UML Community Edition. Not for Commercial Use.

2.1.3 Environment Requirements

- Development language: Java1.4
- Compile target: Java1.4

2.1.4 Package Structure

com.cronos.onlinereview.autoscreening.management

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- Database connection

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

JDBC.

3.2.2 TopCoder Software Component Dependencies:

- Configuration Manager
- DB Connection Factory
- ID Generator

**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 10.0

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines.

3.3.1 Database Connections

Database connections must not be cached within the component. Connections should be created for each operation and closed afterwards.

3.3.2 Component Scalability

The component needs to be scalable. Running multiple instances in the same JVM or in multiple JVM's concurrently should not cause any problem.

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.