Review Statistics Calculator 1.0 Component Specification

1.Design

TopCoder attempts to improve the competition review process and is in need for updating current systems for integrating with new review process outlined in separate Competition Review Process Conceptualization document. Currently there are two applications involved in review process. TopCoder Website application is used by the reviewers for signing up for the review positions and listing the statistics for projects and users. Online Review application is actually used for managing the review process providing the users with abilities to upload submissions for contests, fill the screening and review scorecards, submitting and resolving appeals, performing contest results aggregation, submitting final fixes and approving the results of contests.

This component implements the review statistics collection functionality of the upgrade application.

1.1Design Patterns

Strategy pattern – ReviewerStatisticsCalculator uses pluggable instances of TimelineReliabilityCalculator, CoverageCalculator, AccuracyCalculator, ResourceManager, PhaseManager, DeliverableManager, ReviewManager and ProjectManager; StatisticsCalculator together with its implementation can be used in some external strategy context.

Observer pattern – ReviewerStatisticsCalculator holds a list of registered StatisticsPostCalculationHandler instances and invokes them when reviewer statistics calculation is finished.

1.2Industry Standards

None

1.3Required Algorithms

1.3.1Logging

This component must perform logging in all public methods of ReviewerStatisticsCalculator, TimelineReliabilityCalculatorImpl, CoverageCalculatorImpl and AccuracyCalculatorImpl (except configure() methods).

All information must be logged using log:Log attribute. If this attribute is equal to null, then logging is not required to be performed.

In all mentioned methods method entrance with input arguments, method exit with return value and call duration time must be logged at DEBUG level. It’s not required to log method exit when method throws an exception.

All errors (for all thrown exceptions) must be logged at ERROR level with exception message and stack trace.

1.4Component Class Overview

AccuracyCalculator [interface]

This interface represents an accuracy calculator that can calculate accuracy coefficients for reviewers by the given review scorecards. This interface extends Configurable interface to support configuration via Configuration API component.

AccuracyCalculatorImpl

This class is an implementation of AccuracyCalculator that uses two configurable "evaluationType-accuratePoints" and "evaluationType-inaccuratePoints" mappings and counts points for each review comment. The calculation is performed for each reviewer separately (scorecard of other reviewers are not taken into account). The calculated coefficient equals to (sum of accurate points for reviewer) / (sum of accurate and inaccurate

points for reviewer). Additionally this calculator allows setting the minimum limit for the calculated coefficient value. This class uses Logging Wrapper logger to perform logging of errors and debug information.

BaseCalculator [abstract]

This is a base class for TimelineReliabilityCalculator, CoverageCalculator and AccuracyCalculator implementations provided in this component. It simply holds Logging Wrapper logger and minimum coefficient value to be used by subclasses and performs initialization of these parameters using Configuration API.

Configurable [interface]

This interface must be implemented by all classes that support initialization using Configuration API component.

CoverageCalculator [interface]

This interface represents a coverage calculator that can calculate coverage coefficients for reviewers by the given review scorecards. This interface extends Configurable interface to support configuration via Configuration API component.

CoverageCalculatorImpl

This class is an implementation of CoverageCalculator that uses configurable "evaluationType-points" mapping and counts points for each review comment. The calculated coefficient equals to (sum of points for specific reviewer) / (sum of points for all reviewers). Additionally this calculator allows setting the minimum limit for the calculated coefficient value. This class uses Logging Wrapper logger to perform logging of errors and debug information.

ReviewerStatisticsCalculator

This is the main class of this component. It is an implementation of StatisticsCalculator that extracts all review specific data from persistence using managers and uses aggregated implementations of TimelineReliabilityCalculator, CoverageCalculator and AccuracyCalculator to perform the actual calculation of each reviewer statistics coefficient. ReviewerStatisticsCalculator performs distribution of eligibility points among all secondary reviewers, proportional to their total eligibility points. This class allows to register and unregister the statistics post calculation handlers that are called after each reviewer statistics instance is calculated. This class uses Logging Wrapper logger to perform logging of errors and debug information.

StatisticsCalculator [interface]

This interface represents a reviewer statistics calculator that can calculate reviewer statistics for specified software contests. Additionally it distributes eligibility points pool among all secondary reviewers. This interface extends Configurable interface to support configuration via Configuration API component.

TimelineReliabilityCalculator [interface]

This interface represents a timeline reliability calculator that can calculate timeline reliability coefficients for reviewers. This interface extends Configurable interface to support configuration via Configuration API component.

TimelineReliabilityCalculatorImpl

This class is an implementation of TimelineReliabilityCalculator that uses configurable "penalty per hour" rates for phase types and subtracts the penalty from the maximum timeline reliability coefficient equal to 1. Additionally this calculator allows setting the minimum limit for the calculated coefficient value. This class uses Logging Wrapper logger to perform logging of errors and debug information.

1.5Component Exception Definitions

AccuracyCalculationException

This exception is thrown by ReviewerStatisticsCalculator and implementations of

AccuracyCalculator when some error occurs while performing the calculation of accuracy coefficients.

CoverageCalculationException
This exception is thrown by ReviewerStatisticsCalculator and implementations of CoverageCalculator when some error occurs while performing the calculation of coverage coefficients.

PersistenceException
This exception is thrown by implementations of StatisticsCalculator when some error occurs while accessing the data in persistence or data retrieved from persistence is logically incorrect.

ProjectNotFoundException
This exception is thrown by implementations of StatisticsCalculator when project with the given ID doesn't exist in persistence.

StatisticsCalculatorConfigurationException
This exception is thrown by implementations of Configurable when some error occurs while initializing an instance using the given configuration.

StatisticsCalculatorException
This exception is thrown by implementations of CoverageCalculator, AccuracyCalculator, TimelineReliabilityCalculator and StatisticsCalculator when some unexpected error occurs. Also this exception is used as a base class for other specific custom exceptions.

TimelineReliabilityCalculationException
This exception is thrown by ReviewerStatisticsCalculator and implementations of TimelineReliabilityCalculator when some error occurs while performing the calculation of timeline reliability coefficients.

1.6Thread Safety

This component is thread safe.

Implementations of StatisticsCalculator and TimelineReliabilityCalculator must be thread safe.

Implementations of CoverageCalculator and AccuracyCalculator are not required to be thread safe when entities passed to them are used by the caller in thread safe manner.

Implementations of Configurable are not required to be thread safe.

For all interfaces It's assumed that configure() method will be called just once right after instantiation, before calling any business methods.

ReviewerStatisticsCalculator is immutable and thread safe assuming that configure() method will be called just once right after instantiation, before calling calculateStatistics() method. Since all managers used by this class are not thread safe, the additional synchronization on "this" instance is used when accessing them. The Log and calculator instances used by this class are thread safe.

BaseCalculator is immutable and thread safe assuming that configure() method will be called just once right after instantiation, before calling any business method.

AccuracyCalculatorImpl is immutable and thread safe assuming that configure() method will be called just once right after instantiation, before calling calculateAccuracy() method. The Log instance used by this class is thread safe.

TimelineReliabilityCalculatorImpl is immutable and thread safe assuming that configure() method will be called just once right after instantiation, before calling calculateReliability() method. The Log instance used by this class is thread safe.

CoverageCalculatorImpl is immutable and thread safe assuming that configure() method will be called just once right after instantiation, before calling calculateCoverage() method. The Log instance used by this class is thread safe.

2.Environment Requirements

2.1Environment

Development language: Java1.5
Compile target: Java1.5
QA Environment: Java 1.5, Solaris 7, RedHat Linux 7.1, Windows 2000, Windows 2003, Oracle, JBoss 4.2.2

2.2TopCoder Software Components

Base Exception 2.0 – is used by custom exceptions defined in this component.

Logging Wrapper 1.2 – is used for logging errors and debug information in this component.

Configuration API 1.0 – is used for providing configuration for this component.

Object Factory 2.1.2 – is used for creating pluggable object instances.

Object Factory Configuration API Plugin 1.0 – allows to use Configuration API for creating Object Factory.

Project Management 1.1 – defines ProjectManager and project specific entities used in this component.

Deliverable Management 1.1.1 – defines DeliverableManager and deliverable specific entities used in this component.

Phase Management 1.0.4 – defines PhaseManager used in this component.

Project Phases 2.0.1 – defines Project, Phase and PhaseType entities used in this component.

Resource Management 1.3 – defines ResourceManager, ResourceFilterBuilder and some entities used in this component.

Review Performance Notification 1.0 – defines StatisticsPostCalculationHandler used in this component.

Review Data Structure 1.1 – defines review scorecard specific entities used in this component.

Review Management 1.1 – defines ReviewManager used in this component.

Search Builder 1.4.1 – is used when searching for resources, reviews and deliverables using a filter.

NOTE: The default location for TopCoder Software component jars is../lib/tcs/COMPONENT_NAME/COMPONENT_VERSION relative to the component installation. Setting the tcs_libdir property in topcoder_global.properties will overwrite this default location.

2.3Third Party Components

None

3.Installation and Configuration

3.1Package Name

com.cronos.onlinereview.review.statistics
com.cronos.onlinereview.review.statistics.impl

3.2Configuration Parameters

3.2.1Configuration of ReviewerStatisticsCalculator

The following table describes the structure of ConfigurationObject passed to the configure() method of ReviewerStatisticsCalculator class (angle brackets are used for identifying child configuration objects).

Parameter	Description	Values
<objectFactoryConfig>	This section contains configuration of Object Factory used by this class for creating pluggable object instances.	ConfigurationObject. Required.
loggerName	The logger name passed to LogFactory when creating a Log instance to be used by this class. When not provided, logging is not performed.	String. Not empty. Optional.
timelineReliabilityCalculatorKey	The Object Factory key that is used for creating an instance of TimelineReliabilityCalculator to be used by this class.	String. Not empty. Required.
coverageCalculatorKey	The Object Factory key that is used for creating an instance of CoverageCalculator to be used by this class.	String. Not empty. Required.
accuracyCalculatorKey	The Object Factory key that is used for creating an instance of AccuracyCalculator to be used by this class.	String. Not empty. Required.
timelineReliabilityCalculator Config	The configuration object passed to configure() method of TimelineReliabilityCalculator instance created with object factory.	ConfigurationObject. Required.
coverageCalculatorConfig	The configuration object passed to configure() method of CoverageCalculator instance created with object factory.	ConfigurationObject. Required.
accuracyCalculatorConfig	The configuration object passed to configure() method of AccuracyCalculator instance created with object factory.	ConfigurationObject. Required.
resourceManagerKey	The Object Factory key that is used for creating an instance of ResourceManager to be used by this class.	String. Not empty. Required.
phaseManagerKey	The Object Factory key that is used for creating an instance of PhaseManager to be used by this class.	String. Not empty. Required.
deliverableManagerKey	The Object Factory key that is used for creating an instance of DeliverableManager to be used by this class.	String. Not empty. Required.
reviewManagerKey	The Object Factory key that is used for creating an instance of ReviewManager to be used by this class.	String. Not empty. Required.
projectManagerKey	The Object Factory key that is used for creating an instance of ProjectManager to be used by this class.	String. Not empty. Required.

screeningPhaseTypeId	The ID of the screening phase type.	String representation of positive long integer. Required.
secondaryReviewerReview PhaseTypeId	The ID of the secondary reviewer review phase type.	String representation of positive long integer. Required.
primaryReviewEvaluation PhaseTypeId	The ID of the primary review evaluation phase type.	String representation of positive long integer. Required.
primaryReviewAppeals ResponsePhaseTypeId	The ID of the primary review appeals response phase type.	String representation of positive long integer. Required.
aggregationReview PhaseTypeId	The ID of the aggregation review phase type.	String representation of positive long integer. Required.
finalReviewPhaseTypeId	The ID of the final review phase type.	String representation of positive long integer. Required.
secondaryReviewer ResourceRoleId	The ID of the secondary reviewer resource role.	String representation of positive long integer. Required.
primaryReviewEvaluator ResourceRoleId	The ID of the primary review evaluator resource role.	String representation of positive long integer. Required.

3.2.2Configuration of BaseCalculator subclasses

The following table describes the structure of ConfigurationObject passed to the configure() method of BaseCalculator class (angle brackets are used for identifying child configuration objects).

Parameter	**Description**	**Values**
loggerName	The logger name passed to LogFactory when creating a Log instance to be used by this class. When not provided, logging is not performed.	String. Not empty. Optional.
minimumCoefficient	The minimum coefficient value that can be generated by this calculator. Default is “0”.	String representation of not negative double. Optional.

3.2.3Configuration of TimelineReliabilityCalculatorImpl

The following table describes the structure of ConfigurationObject passed to the configure() method of TimelineReliabilityCalculatorImpl class (angle brackets are used for identifying child configuration objects).

Parameter	**Description**	**Values**

penaltyPerHourFor PhaseTypeX	“X” here is any positive integer number representing the phase type ID. The value is the penalty per hour for phases of this type. This penalty represents the value deducted from the timeline reliability coefficient when reviewer's delay equals to 1 hour.	String representation of not negative double. Multiple. Optional.
defaultPenaltyPerHour	The penalty per hour for all phase types not mentioned with penaltyPerHourForPhaseTypeX parameters. Default is “0” (no penalty).	String representation of not negative double. Optional.

Please see section 3.2.2 for additional parameters to be specified in the configuration of TimelineReliabilityCalculatorImpl.

3.2.4Configuration of CoverageCalculatorImpl

The following table describes the structure of ConfigurationObject passed to the configure() method of CoverageCalculatorImpl class (angle brackets are used for identifying child configuration objects).

Parameter	**Description**	**Values**
pointsForEvaluationTypeX	“X” here is any positive integer number representing the phase type ID. The number of points (weight) associated with comments that have the specified evaluation type.	String representation of double. Multiple. At least one is required.

Please see section 3.2.2 for additional parameters to be specified in the configuration of CoverageCalculatorImpl.

3.2.5Configuration of AccuracyCalculatorImpl

The following table describes the structure of ConfigurationObject passed to the configure() method of AccuracyCalculatorImpl class (angle brackets are used for identifying child configuration objects).

Parameter	**Description**	**Values**
pointsForAccurate EvaluationTypeX	“X” here is any positive integer number representing the phase type ID. The number of points (weight) associated with comments that have the specified evaluation type. Comments with this evaluation type are assumed to be correct.	String representation of not negative double. Multiple. At least one is required.
pointsForInaccurate EvaluationTypeX	“X” here is any positive integer number representing the phase type ID. The number of points (weight) associated with comments that have the specified evaluation type. Comments with this evaluation type are assumed to be erroneous.	String representation of not negative double. Multiple. At least one is required.

Please see section 3.2.2 for additional parameters to be specified in the configuration of AccuracyCalculatorImpl.

3.3Dependencies Configuration

Please see docs of dependency components to configure them properly.

4.Usage Notes

4.1Required steps to test the component

- Extract the component distribution.
- Follow .
- Execute ‘ant test’ within the directory that the distribution was extracted to.

4.2Required steps to use the component

Please see the demo.

4.3Demo

4.3.1API usage

```
// ConfigManager
ConfigManager cm = ConfigManager.getInstance();

// load config files.
cm.add(new File(TEST_PATH + "ConnectionFactory.xml").getAbsolutePath());
cm.add(new File(TEST_PATH + "InformixPersistence.xml").getAbsolutePath());
cm.add(new File(TEST_PATH + "PhaseManager.xml").getAbsolutePath());
cm.add(new File(TEST_PATH + "ProjectManager.xml").getAbsolutePath());
cm.add(new File(TEST_PATH + "ReviewManager.xml").getAbsolutePath());
cm.add(new File(TEST_PATH + "SearchBundleManager.xml").getAbsolutePath());

// Create an instance of ReviewerStatisticsCalculator
ReviewerStatisticsCalculator reviewerStatisticsCalculator = new
ReviewerStatisticsCalculator();

// Get configuration for ReviewerStatisticsCalculator
ConfigurationObject config = TestsHelper.loadConfig("config.xml").getChild(
            "com.cronos.onlinereview.review.statistics.impl.ReviewerStatisticsCalculator");

// Configure the calculator
reviewerStatisticsCalculator.configure(config);

// Register post calculation handler
StatisticsPostCalculationHandler handler =
EasyMock.createMock(StatisticsPostCalculationHandler.class);
reviewerStatisticsCalculator.registerPostCalculationHandler(handler);

// Unregister this post calculation handler
reviewerStatisticsCalculator.unregisterPostCalculationHandler(handler);

// Calculate statistics for project with ID=1,
// distribute 2 eligibility points among secondary reviewers
ReviewerStatistics[] reviewerStatistics =
reviewerStatisticsCalculator.calculateStatistics(1L, 2);

// Check results
assertEquals(3, reviewerStatistics.length);

double value = 1.0 - (2.5 * 0.04 + 1.2 * 0.02 + 3.5 * 0.02);
assertEquals(value, reviewerStatistics[0].getTimelineReliability());
assertEquals(-1.0, reviewerStatistics[0].getAccuracy());
assertEquals(-1.0, reviewerStatistics[0].getCoverage());

assertEquals(0.95, reviewerStatistics[1].getTimelineReliability());
assertEquals(32.0 / 62.0, reviewerStatistics[1].getAccuracy());
assertEquals(32.0 / 78.0, reviewerStatistics[1].getCoverage());

assertEquals(1.0, reviewerStatistics[2].getTimelineReliability());
assertEquals(46.0 / 86.0, reviewerStatistics[2].getAccuracy());
assertEquals(46.0 / 78.0, reviewerStatistics[2].getCoverage());
```

```
    // Clear ConfigManager
    for (Iterator iterator = cm.getAllNamespaces(); iterator.hasNext();) {
           String namespace = (String) iterator.next();
           if ("com.topcoder.util.log".equals(namespace)) {
               continue;
           }
           cm.removeNamespace(namespace);
    }
```

4.3.2 Sample ReviewerStatisticsCalculator configuration

This XML configuration can be read to ConfigurationObject instance using Configuration Persistence component. Please see its docs for details.

```
<?xml version="1.0"?>
<CMConfig>
   <Config name="com.topcoder.db.connectionfactory.DBConnectionFactoryImpl">
      <Property name="connections">
         <!-- the "default" property refers to a configured connection -->
         <Property name="default">
            <Value>sysuser</Value>
         </Property>
         <Property name="sysuser">
            <Property name="producer">
               <Value>com.topcoder.db.connectionfactory.producers.JDBCConnectionProducer</Value>
            </Property>
               <Property name="parameters">
               <Property name="jdbc_driver">
                  <Value>com.informix.jdbc.IfxDriver</Value>
               </Property>
               <Property name="jdbc_url">
                     <Value>jdbc:informix-
sqli://127.0.0.1:9088/topcoder:INFORMIXSERVER=demo</Value>
               </Property>
               <Property name="user">
                     <Value>informix</Value>
               </Property>
               <Property name="password">
                  <Value>topcoder</Value>
               </Property>
            </Property>
         </Property>
      </Property>
   </Config>
   <Config name="com.cronos.onlinereview.review.statistics.impl.ReviewerStatisticsCalculator">
      <Property name="objectFactoryConfig">
         <Property name="timelineReliabilityCalculator">
            <Property name="type">

 <Value>com.cronos.onlinereview.review.statistics.impl.TimelineReliabilityCalculatorImpl</Value>
            </Property>
         </Property>
         <Property name="coverageCalculator">
            <Property name="type">
               <Value>com.cronos.onlinereview.review.statistics.impl.CoverageCalculatorImpl</Value>
            </Property>
         </Property>
         <Property name="accuracyCalculator">
            <Property name="type">
               <Value>com.cronos.onlinereview.review.statistics.impl.AccuracyCalculatorImpl</Value>
            </Property>
         </Property>
         <Property name="resourceManager">
            <Property name="type">

 <Value>com.topcoder.management.resource.persistence.PersistenceResourceManager</Value>
            </Property>
            <Property name="params">
               <Property name="param1">
                  <Property name="name">
```

```
                        <Value>persistence</Value>
                     </Property>
                  </Property>
                  <Property name="param2">
                     <Property name="name">
                        <Value>searchBundleManager</Value>
                     </Property>
                  </Property>
               </Property>
            </Property>
            <Property name="persistence">
               <Property name="type">

<Value>com.topcoder.management.resource.persistence.sql.SqlResourcePersistence</Value>
               </Property>
               <Property name="params">
                  <Property name="param1">
                     <Property name="type">
                        <Value>com.topcoder.db.connectionfactory.DBConnectionFactoryImpl</Value>
                     </Property>
                  </Property>
               </Property>
            </Property>
            <Property name="searchBundleManager">
               <Property name="type">
                  <Value>com.topcoder.search.builder.SearchBundleManager</Value>
               </Property>
               <Property name="params">
                     <Property name="param1">
                        <Property name="type">
                                <Value>String</Value>
                        </Property>
                        <Property name="value">
                         <Value>com.topcoder.search.builder.SearchBundleManager</Value>
                        </Property>
                     </Property>
               </Property>
            </Property>
            <Property name="phaseManager">
               <Property name="type">
                  <Value>com.topcoder.management.phase.DefaultPhaseManager</Value>
               </Property>
               <Property name="params">
                     <Property name="param1">
                        <Property name="type">
                        <Value>String</Value>
                        </Property>
                        <Property name="value">
                         <Value>com.topcoder.management.phase.Default</Value>
                        </Property>
                     </Property>
               </Property>
            </Property>
            <Property name="deliverablePersistence">
               <Property name="type">
               <Value>com.topcoder.management.deliverable.persistence.sql.SqlDeliverablePersistence
               </Value>
               </Property>
               <Property name="params">
                  <Property name="param1">
                     <Property name="name">
                        <Value>dbFactory</Value>
                     </Property>
                  </Property>
               </Property>
            </Property>
            <Property name="dbFactory">
               <Property name="type">
                  <Value>com.topcoder.db.connectionfactory.DBConnectionFactoryImpl</Value>
```

```
      </Property>
      <Property name="params">
         <Property name="param1">
            <Property name="type">
               <Value>String</Value>
            </Property>
            <Property name="value">
               <Value>com.topcoder.db.connectionfactory.DBConnectionFactoryImpl</Value>
            </Property>
         </Property>
      </Property>
   </Property>
   <Property name="deliverableManager">
      <Property name="type">
         <Value>com.topcoder.management.deliverable.PersistenceDeliverableManager</Value>
      </Property>
      <Property name="params">
            <Property name="param1">
               <Property name="name">
                        <Value>deliverablePersistence</Value>
               </Property>
            </Property>
            <Property name="param3">
               <Property name="name">
                <Value>searchBundleManager</Value>
               </Property>
         </Property>
      </Property>
   </Property>
   <Property name="reviewManager">
      <Property name="type">
         <Value>com.topcoder.management.review.DefaultReviewManager</Value>
      </Property>
   </Property>
   <Property name="projectManager">
      <Property name="type">
         <Value>com.topcoder.management.project.ProjectManagerImpl</Value>
      </Property>
   </Property>
</Property>

<Property name="loggerName">
   <Value>myLogger</Value>
</Property>
<Property name="minimumCoefficient">
   <Value>0.8</Value>
</Property>
<Property name="timelineReliabilityCalculatorKey">
   <Value>timelineReliabilityCalculator</Value>
</Property>
<Property name="coverageCalculatorKey">
   <Value>coverageCalculator</Value>
</Property>
<Property name="accuracyCalculatorKey">
   <Value>accuracyCalculator</Value>
</Property>
<Property name="resourceManagerKey">
   <Value>resourceManager</Value>
</Property>
<Property name="phaseManagerKey">
      <Value>phaseManager</Value>
</Property>
<Property name="deliverableManagerKey">
   <Value>deliverableManager</Value>
</Property>
<Property name="reviewManagerKey">
   <Value>reviewManager</Value>
</Property>
<Property name="projectManagerKey">
```

```
      <Value>projectManager</Value>
   </Property>
   <Property name="screeningPhaseTypeId">
      <Value>3</Value>
   </Property>
   <Property name="secondaryReviewerReviewPhaseTypeId">
      <Value>13</Value>
   </Property>
   <Property name="primaryReviewEvaluationPhaseTypeId">
      <Value>14</Value>
   </Property>
   <Property name="primaryReviewAppealsResponsePhaseTypeId">
      <Value>17</Value>
   </Property>
   <Property name="aggregationReviewPhaseTypeId">
      <Value>18</Value>
   </Property>
   <Property name="finalReviewPhaseTypeId">
      <Value>10</Value>
   </Property>
   <Property name="secondaryReviewerResourceRoleId">
      <Value>16</Value>
   </Property>
   <Property name="primaryReviewEvaluatorResourceRoleId">
      <Value>17</Value>
   </Property>
   <Property name="timelineReliabilityCalculatorConfig">
      <Property name="loggerName">
         <Value>myLogger</Value>
      </Property>
      <Property name="minimumCoefficient">
         <Value>0.5</Value>
      </Property>
      <Property name="defaultPenaltyPerHour">
         <Value>0.02</Value>
      </Property>
      <Property name="penaltyPerHourForPhaseType3">
         <Value>0.04</Value>
      </Property>
   </Property>
   <Property name="coverageCalculatorConfig">
      <Property name="loggerName">
         <Value>myLogger</Value>
      </Property>
      <Property name="minimumCoefficient">
         <Value>0.1</Value>
      </Property>
      <Property name="pointsForEvaluationType1">
         <Value>10</Value>
      </Property>
      <Property name="pointsForEvaluationType2">
         <Value>3</Value>
      </Property>
      <Property name="pointsForEvaluationType3">
         <Value>1</Value>
      </Property>
   </Property>
   <Property name="accuracyCalculatorConfig">
      <Property name="loggerName">
         <Value>myLogger</Value>
      </Property>
      <Property name="minimumCoefficient">
         <Value>0.1</Value>
      </Property>
      <Property name="pointsForAccurateEvaluationType1">
         <Value>10</Value>
      </Property>
      <Property name="pointsForAccurateEvaluationType2">
         <Value>3</Value>
```

```
            </Property>
            <Property name="pointsForAccurateEvaluationType3">
               <Value>1</Value>
            </Property>
            <Property name="pointsForInaccurateEvaluationType6">
               <Value>10</Value>
            </Property>
         </Property>
      </Config>
   </CMConfig>
```

4.3.3Calculation sample

Assume that configuration provided in the section 4.3.2 is used.

Assume that the following evaluation types are persisted:

Evaluation type ID	**Description**
1	Serious
2	Medium
3	Minor
4	Comment
5	Combined
6	Incorrect

Also assume that the project for which reviewer statistics are calculated have 2 submissions that passed screening (S1 and S2).

Assume that the review is performed by secondary reviewers A and B. Evaluation is performed by the primary reviewer C.

The table below describes the number of comment reviewers A and B specified in the scorecard respective to the evaluation type specified for these comments by the primary reviewer:

Secondary reviewer	**Submission**	**Number of comments with specific evaluation type ID**					
		1	**2**	**3**	**4**	**5**	**6**
A	S1	1	2	4	0	6	2
	S2	0	2	1	1	4	1
	S3	0	1	2	0	1	0
	S1 + S2 + S3	*1*	*5*	*7*	*1*	*11*	*3*
B	S1	1	1	1	1	5	3
	S2	1	2	4	0	3	1
	S3	1	0	2	2	3	0
	S1 + S2 + S3	*3*	*3*	*7*	*3*	*11*	*4*

The following table specifies the number of hours for each (reviewer; phase) that represent how much the reviewer didn't meet the timeline (was late when submitting a delivery):

Reviewer	**Delay in hours per phase (phase ID)**

	Screening (3)	Secondary Reviewer Review (13)	Primary Review Evaluation (14)	Primary Review Appeals Response (17)
A	-	2.5	-	-
B	-	0	-	-
C	2.5	-	1.2	3.5

Note that hours are taken here for simplicity only; actually the difference between deadline and commitment time is calculated with the millisecond accuracy.

The timeline reliability coefficient calculation will be the following:

Reviewer	Calculation	Result
A	1 – 2.5 * 0.02	0.95
B	1	1
C	1 – (2.5 * 0.04 + 1.2 * 0.02 + 3.5 * 0.02)	0.806

The coverage coefficient calculation will be the following:

Reviewer	Calculation of row score	Row score	Sum of row scores	Calculation	Result
A	1 * 10 + 5 * 3 + 7 * 1	32	32 + 46 = 78	32 / 78	0.410
B	3 * 10 + 3 * 3 + 7 * 1	46		46 / 78	0.590

The accuracy coefficient calculation will be the following:

Reviewer	Accurate points	Inaccurate points	Calculation	Result
A	1 * 10 + 5 * 3 + 7 * 1 = 32	3 * 10 = 30	32 / (32 + 30)	0.516
B	3 * 10 + 3 * 3 + 7 * 1 = 46	4 * 10 = 40	46 / (46 + 40)	0.535

The total evaluation coefficients will be the following:

Reviewer	Calculation	Result
A	0.95 * 0.410 * 0.516	0.200
B	1 * 0.590 * 0.535	0.316
C	0.806	0.806

The distribution of eligibility points will be the following:

Reviewer	eligibilityPointsPool	Calculation	Result
A	2	2 * 0.2 / (0.2 + 0.316)	0.775
B		2 * 0.316 / (0.2 + 0.316)	1.225

5.Future Enhancements

None