Requirements Specification

1. Scope

1.1 Overview

The Math Matrix component is designed to give developers the ability to set up and evaluate complex matrices. The matrix provides percentages to evaluate a score based on the total score. The component also handles weighted calculations by line item and groups of line items. This allows a developer to give a higher or lower weight to different parts of the matrix.

1.2 Logic Requirements

1.2.1 Calculations

- Calculate a total weighted value for a set of line items.
- Line items can be broken up into a group. Each group can be assigned a weighted percentage of the total. The sum of all group percentages is 100%.
- Each line item within a group will also be assigned a weighted percentage. The sum of all line item percentages is 100%.
- The total allowable score can be configurable. For example, if the sum of group totals is 100 out of 200 points, the user must be able to represent that out of 100, 200 or 1000.
- NOTE: This component is not intended to interact with a spreadsheet application. The component should accept all parameters through its API.

Example:

	Weight	Maximum Value	Actual Score	Weighted Score
Section 1	**25%**			
Requirement 1	0.5	4	4	12.50
Requirement 2	0.15	4	4	3.75
Requirement 3	0.15	4	4	3.75
Requirement 4	0.2	4	4	5.00
	100.00%		16	25.00
Section 2	**20%**			
Requirement 1	0.5	4	4	10.00
Requirement 2	0.5	4	4	10.00
	100.00%		8	20
Section 3	**15%**			
Requirement 1	1	4	4	15.00
	100.00%		4	15
Section 4	**40%**			
Requirement 1	0.33	4	4	13.20
Requirement 2	0.33	4	4	13.20
Requirement 3	0.34	4	4	13.60
	100.00%		12	40
Total Weighting	**100.00%**			
Total Possible Score	**100**			***100***

1.3 Example of the Software Usage

An example usage for the complex matrix is the TopCoder Software review scorecards used to evaluate component design and development projects. The scorecards have a series of questions that are grouped by subject area. Each subject area has a weight assigned to it. Each question is a percentage total weight for that group which is a percentage of the total weight for the scorecard.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java1.4
- Compile target: Java1.2, Java1.3, Java1.4

2.1.4 Package Structure

Com.topcoder.math.matrix

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 Third Party Component, Library, or Product Dependencies:

None.

3.2.3 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000

3.3 Design Constraints

3.3.1 Development Standards:

- It is required that all code be clearly commented.
- All code must adhere to javadoc standards, and, at minimum, include the @author, @param, @return, @throws and @version tags.
  - When populating the author tag, use your TopCoder member handle. In addition, please do not put in your email addresses.
  - Copyright tag: Copyright © 2002, TopCoder, Inc. All rights reserved
- For standardization purposes, code must use a 4-space (not tab) indentation.
- Do not use the ConfigurationManager inside of EJB's. The usage of the java.io.* package to read/write configuration files can potentially conflict with a restrictive security scheme inside the EJB container.
- All code must adhere to the Code Conventions outlined in the following: http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html

3.3.2 Database Standards:

- Table and column names should be as follows: table_name, not tableName.
- Every table must have a Primary Key.
- Always use Long (or database equivalent) for the Primary Key.
  For maximum portability, database objects (table names, indexes, etc) should be kept to 18 characters or less. Informix and DB2 have an 18-character limit.