Software Documentation : Java Generic Digital Run Entities

This page last changed on Jun 13, 2008 by pulky.

1. Scope

1.1 Overview

This component defines entities to represent database information, providing also the O/R mapping.
A DB diagram (DataModel.png) and DDL for creating the DB tables (DRSchema.ddl) are provided together with this specification.

1.1.1 Version

1.0

1.2 Logic Requirements

1.2.1 Entities and mapping

All the tables in the diagram must be fully supported. If you consider that changes are needed to the provided database design, please ask in the forums first.
Notice that create_date and modify_date fields present in several tables should also be mapped to entities, but they should work as read only attributes since they will be automatically filled by the database.
All the tables with a key identifier must generate its unique key using Id Generator component. The only immutable table is project_type_lu.
The mapping must be done in one or more XML files using JPA with Hibernate extensions.

1.2.1.1 track table

Each row in this table represents a digital run track like "2007.5 Design cup".

Column name	Purpose	Sample values
track_desc	Stores the track's description	2008 July Design cup
track_start_date	Stores the track's start date	07/01/2008
track_end_date	Stores the track's start date	07/31/2008

Relationships:

- Track type: defines the type of the track. (Design / Development / Studio / Etc.)
- Track Status: defines the status of the track. (Active / Finalized / Etc.)
- Points calculator: defines the class that can understand the track's rules and calculate the corresponding DR points.
- Project types: each track can have multiple project types associated, this way we can make different project types count for the same track. An example of this would be the Development track including development, component testing, assemblies, etc.
- Contests: each track can have one or more contests associated. Examples for this are Top performers contest, Top N contest.
- DR points: the awarded dr points. (this relationship should be lazy)

1.2.1.2 track_type_lu table

This lookup table stores the track types.

1.2.1.3 track_status_lu table

This lookup table stores the track status.

1.2.1.4 points_calculator_lu table

This lookup table stores the points calculators.

1.2.1.5 track_project_type_xref table

This xref table stores the n:m relationship.

1.2.1.6 project_type_lu table

This lookup table stores the project types.

1.2.1.7 track_contest table

Each row in this table represents a digital run track contest like "Top Performers".

Column name	Purpose	Sample values
track_contest_desc	Stores the track contests's description	Top Performers

Relationships:

- Track: defines the track where this contest belongs to.
- Track contest type: defines the type of the track contest. (Top performers / Top N / Trip winners / Etc.)
- Track contest result calculator: defines the class that can understand the track contest's rules and calculate the corresponding results.

1.2.1.8 track_contest_type_lu table

This lookup table stores the track contest types.

1.2.1.9 track_contest_result_calculator_lu table

This lookup table stores the result calculators.

1.2.1.10 dr_points table

Each row in this table represents digital run points like 500 points for 1st winning submission on project X or %15 dr points bonus for project X because of high reliability.

Column name	Purpose	Sample values
dr_points_desc	Stores the description of the points	Points awarded for 1st place submission for project X.
user_id	Stores the user's id	99999
amount	Stores the amount (the unit will be interpreted differently depending on the points type) Negative values will be interpreted as penalizations.	5000 (points) / 15 (%)

application_date	Stores the date used to determine the corresponding track.	07/5/2008
award_date	Stores the date when the points were awarded	07/15/2008
reference_id	This value references the origin of the points. It can point to different entities (the reference type will define which) Sample entities are projects, another dr points row (for bonuses or penalties), etc.	99999
is_potential	This Boolean value defines if the awarded points are potential or definitive.	True/false

Relationships:

- Track: defines the track where these points belong to.
- Points type: defines the type of the awarded dr points. The type can be DR project points, Reliability bonus, DR penalty, etc.
- Points status: defines the status of the awarded dr points. (Active / Etc.)
- Points operation: defines the operation for the points. The points can be a fixed amount or a percentage
- Points reference type: this defines the type of the reference. The points can refer to a project to another points entry.

1.2.1.11 dr_points_type_lu table

This lookup table stores the dr points types.

1.2.1.12 dr_points_status_lu table

This lookup table stores the dr points status.

1.2.1.13 dr_points_operation_lu table

This lookup table stores the dr points operations.

1.2.1.14 dr_points_reference_type_lu table

This lookup table stores the dr points reference types.

1.2.2 Auditing

The component must audit creation, update and deletion of entities using Auditor 2.0 component. Each audit will contain information about who performed it, when it was performed and the information that was modified (before and after values).
The auditing data will occur in their own separate tables, and those tables will be read only from within the application.
The designer must provide the DDL for the auditing tables.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

Other components will rely on the entities provided by this component to provide higher level services concerning the digital run.

1.5 Future Component Direction

None.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java 1.6
- Compile target: Java 1.6

2.1.4 Package Structure

com.topcoder.service.digitalrun.entity

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?

- JPA (with Hibernate extensions)

3.2.2 TopCoder Software Component Dependencies:

- Id Generator 3.0
- Auditor 2.0

**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 10.00.UC 5
- JBoss 4.2 GA
- Java Persistence API (JPA)
- Hibernate 3.2.5

3.2.4 QA Environment:

- RedHat Enterprise Linux 4
- JBoss 4.2 GA
- Java 1.5
- Informix 10.00.UC 5

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
- DDL for the auditing tables
- O/R mappings

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the 'Documentation' tab.