Software Documentation : Java Custom Late Deliverables Management

This page last changed on Nov 19, 2010 by volodymyrk.

1. Scope

1.1 Overview

TopCoder has an utility called Late Deliverables Tracker that periodically examines all active projects in the Online Review and records when the deliverables are late (e.g. review scorecards, final fixes etc.). Late Deliverables Management provides search and update functionality for the late deliverables. Different search filters are supported such as filtering by the project status, project category, deliverable type etc. It is also possible to combine the filters with AND and OR logical operators.

1.2 Logic Requirements

1.2.1 Late Deliverable Entity

Late deliverables are stored in the tcs_catalog.late_deliverable table. The DDL for the table is provided within the DB schema.

The component will provide an entity class for the late deliverable. This class will include all the fields from the corresponding table. All the search and update methods of the component will work with this class.

1.2.2 Update operation

The component will provide an update method for late deliverables. This method will update the existing late deliverables but it won't create new ones.

1.2.3 Retrieve operation

The component will provide a retrieve method that will return the late deliverable by its ID.

1.2.3 Search operation

The component will provide two methods for searching late deliverables. See sections 1.2.3.1 and 1.2.3.2 for the details.
Both methods will take Filter object which defines the search constraints for the late deliverables. Search Builder component will be used to work with the search filters.

> **Search Builder**
>
> The requirements below assume the designers and developers are familiar with the Search Builder component and the way it is designed. If you are not please go ahead and familiarize yourself with the Search Builder component before proceeding.

1.2.3.1 Search All Late Deliverables

The component will provide the following method:

```
public List<LateDeliverable> searchAllLateDeliverables(Filter filter);
```

The method will return all late deliverables constrained by the specified filter.

1.2.3.2 Search Restricted Late Deliverables

The component will provide the following method:

```
public List<LateDeliverable> searchRestrictedLateDeliverables(Filter filter, long userId);
```

The method will return all late deliverables to which the specified user has access to constrained by the specified filter.

A user has access to a late deliverable if an only if the late deliverable is assigned to this user or if the user has manager access to the respective Online Review project. The manager access can be of two types: a user can either be one of the manager resources for the project (i.e. Manager, Copilot or Client Manager) or the user can have access to the associated cockpit project.

Query to get the list of the project IDs the user has one of the manager roles in is below:

```
select distinct p.project_id
from project p, resource r, resource_info ri
where
p.project_id = r.project_id and
r.resource_id = ri.resource_id and
ri.resource_info_type_id = 1 and
ri.value = USERID and
r.resource_role_id in (13,14,15);
```

Query to get the list of the project IDs the user has "cockpit project access" to is below:

```
select distinct p.project_id
from  project p, corporate_oltp:tc_direct_project d
where p.tc_direct_project_id = d.project_id
and d.project_id in
(
select distinct tdp.project_id
from corporate_oltp:user_permission_grant g,
user u,
tc_direct_project tdp
where is_studio = 0
and u.user_id = g.user_id
and g.resource_id = tdp.project_id
and u.user_id = USERID
)
```

Please note that due to the way the Search Builder component is designed the nested queries (like in the fragments above) are not supported. It is the responsibility of the designer to write the search bundle's context in a way that will join the necessary tables and have all the access constraints above in the context's where clause. Since the two SQL fragments have USERID parameter, which can't be passed to the search bundle's context, the proper constraint should be added as an additional EQUALS filter composed with the Filter parameter of the method.

So, the method will basically perform like the following:

1. Create EQUALS filter for the user ID
2. Compose the EQUALS filter with the filter passed to the method (if not null) using AND filter.
3. Call search bundle's search method with the resulting composed filter.

Providing the configuration for the Search Builder component is an essential part of the design since it contains the bundle contexts which is to be written by the designer.

1.2.4 Search filters

The component will provide factory methods to create the filters defined in sections 1.2.4.1-1.2.4.2.

1.2.4.1 Project filters

The following search filters will be provided by the component:

1. Filter by specific project ID.
2. Filter by project status.
3. Filter by project category.

1.2.4.2 Deliverable filters

The following search filters will be provided by the component:

1. Filter by the deliverable type.
2. Filter by forgiven / unforgiven late deliverables.

1.2.4.3 Composite filters

The component will support composite AND, OR and NOT filters of any of the filters defined in sections 1.2.4.1 and 1.2.4.2. These filters are already provided by the Search Builder component so the design just needs to make sure they are supported.

1.2.5 Thread-Safety

The component is required to be thread-safe.

1.2.6 Performance

The DB will store large number of late deliverables (up to 100 000 and more), projects (up to 100 000 and more) and resources (up to 1 000 000 and more). The component will be used to generate the reports of the late deliverables for the past projects in real-time. Considering that some of the users (mostly TopCoder managers) have access to a large number of past projects the result can be very large depending on the search filters used. Therefore special care should be taken to make sure there are no unnecessary performance overhead and the SQL queries generated by the component are optimized where possible (i.e. proper joins are used instead of nested queries etc.).

1.3 Required Algorithms

Designers must describe all of the algorithms used to construct the queries and perform the search and update operations.

1.4 Example of the Software Usage

The component will be used by the Online Review application to generate reports of the late deliverables with a set of custom filters.

1.6 Future Component Direction

Any enhancement needs to be approved either in forum or in email with managers to eliminate over-complicating the component with useless functions. All performance optimizations are highly encouraged and do not require explicit approval.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None, only API interface will be provided.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java 1.5
- Compile target: Java 1.5, Java 1.6

2.1.4 Package Structure

com.topcoder.management.deliverable.late

com.topcoder.management.deliverable.late.search

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 TopCoder Software Component Dependencies:

- Base Exception 2.0
- Configuration API 1.0
- Configuration Persistence 1.0.2
- Project Management 1.0.1 (optional)
- Project Management Persistence 1.1.2 (optional)
- Phase Management 1.0.4 (optional)
- Phase Management Persistence 1.0.2 (optional)
- Deliverable Management 1.1.1 (optional)
- Deliverable Management Persistence 1.1.1 (optional)
- Online Review Deliverables 1.0.2 (optional)
- Search Builder 1.3.1
- DB Connection Factory 1.1.0
- Logging Wrapper 1.2.0

**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:

Any third party library needs to be approved.

3.2.4 QA Environment:

- Java 1.5
- RedHat Linux 4
- Windows 2000
- Windows 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
- Configuration Files for the Search Builder component

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the 'Documentation' tab of TC UML Tool.