Name: *Change Activity Custom Workflow Assignments*

By: Bob Lach

Change History:

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Description** |
| 12/21/2012 | Bob Lach | Initial version. |

# Best Practice Name and Classification

## Name

Change Activity Custom Workflow Assignments Best Practice

## Classifications

Low level – customization.

# Objective

## Problem Statement

Need to customize the change activity workflow template so that rework activities are assigned to the users that actually completed the original activities, rather than simply being reassigned to the whole team.

## Background

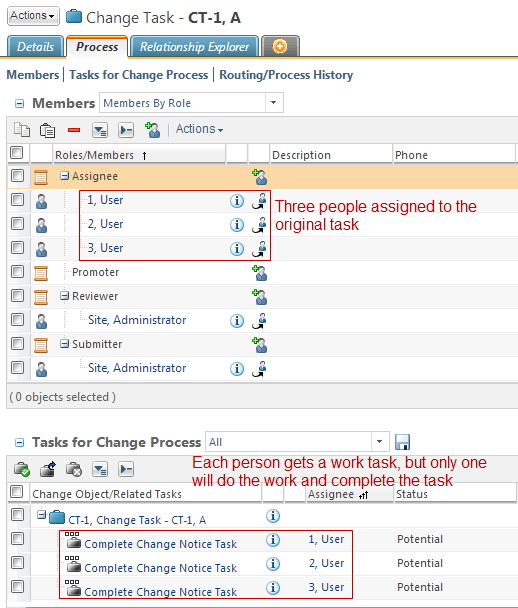
The Windchill out-of-the-box (OOTB) change activity workflow template will assign rework activities to all users specified in the “Assignee” role, regardless of which user(s) actually completed the original assignment. While this is appropriate for a general OOTB workflow template, it may not serve the needs of customers who assign groups of people to change activities and allow anyone in the group to complete the task. In this case it may be desirable to assign rework activity to the person who actually did the original task so they can quickly correct the work, rather than assign it to the whole group where someone unfamiliar with the original work may pick it up. In addition, a customer may wish to use a different team role for rework activities, such as a “Rework Assignee”.

## Scope/Applicability/Assumptions

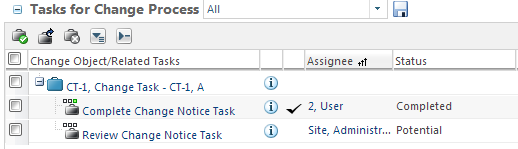
This document assumes familiarity with workflow processes for change objects and workflow customization. The techniques described in this document are limited to change object workflow templates.

## Intended Outcome

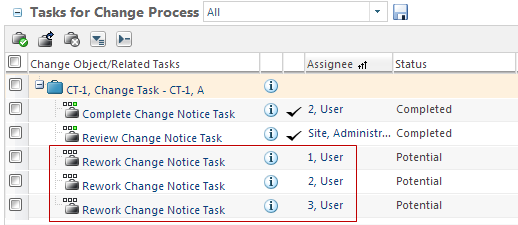
Assume a case where a change task is created with three assigned participants using the OOTB Change Activity Workflow template. When the change task is first created its process page will show the following.



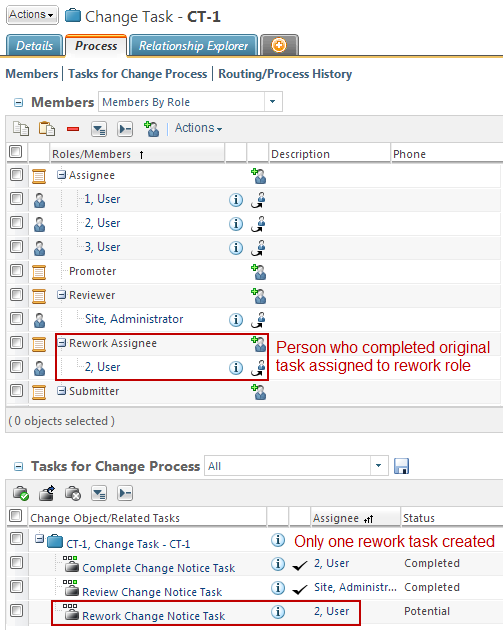
If User 2 actually takes on the task and completes it, then the Tasks for Change Process table will be updated to show the completed work task and a review task.



If a problem is found during review, and the task is sent for rework, then a new rework task will be created for each of the original assignees. Any one of them may now accept the rework task.



Creating rework tasks for all original team members may run contrary to a company’s standard change process. For example, a change process may dictate that only User 2, who originally performed the work, should be assigned to a special “rework” role. And further, User 2 should be the only person assigned to a rework task. The desired process page should show the following. This is only possible by manually editing the process when using the OOTB Change Activity Workflow template. A customized template is needed to automatically follow this change process.



# Solution

## Solution Statement

Customize the Change Activity Workflow template to automatically determine who completed a work task and assign only that person (or persons) to the rework role.

## Prerequisite knowledge

To apply this best practice, you need to have an understanding of the following:

* Workflow Processes for Change Objects
* Workflow Template Administration and Customization
* Life Cycle Template Administration and Customization
* Object Initialization Rules Administration and Customization
* Enumerated Type Customization
* Basic Java programming

The [See Also](#_See_Also) section below includes references to many or all of these subjects.

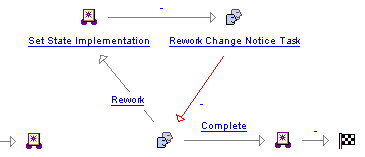
# Customization Points

## Create Custom Rework Role

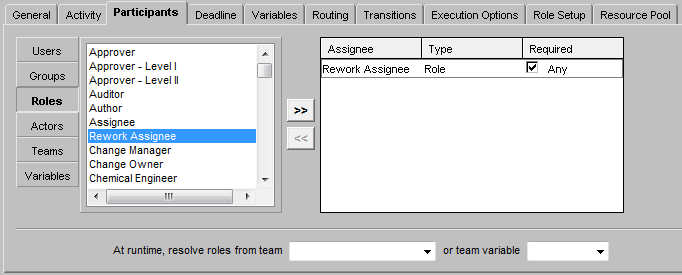
Use the enumCustomize tool to add a new rework role to wt.project.RoleRB.rbInfo. See the Windchill Customizer’s Guide for instructions on using this tool. Custom role “Rework Assignee” will be used in this document. This step can be skipped if a new role is not needed.

## Customize Change Activity Workflow Template

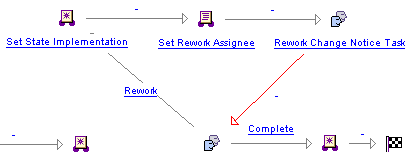
1. Select Workflow Template Administration from Utilities page of Site, Organization, Product or Library context.
2. Select the “Save As” action for “Change Activity Workflow” template to create a copy for customization. Select suitable name for the customized workflow (e.g., “Custom Change Activity Workflow”) will be used in this document. Always copy a workflow template before changing it; always avoid customizing an OOTB workflow template.
3. Select the “Edit” action for “Custom Change Activity Workflow” template and the change activity workflow graph is displayed in Workflow Template Editor. The area of interest for this customization is the rework loop, located in the upper-right portion.



1. Double-click on the “Rework Change Notice Task” to open the editor and select the “Participants” tab. Add the “Rework Assignee” role and remove the “Assignee” role. This step can be skipped if a new role is not needed.



1. Create a new expression step in the workflow between the “Set State Implementation” and “Rework Change Notice Task” steps, and name it something meaningful, such as “Set Rework Assignee”.



1. Add Java code to the “Set Rework Assignee” step to automatically obtain the user(s) that completed the prior work task and assign them to the change item team for the rework task. See section 5 for a description of helper methods that can be called here.  
     
   Example:

wt.project.Role reworkRole = wt.project.Role.toRole("REWORK\_ASSIGNEE");

wt.project.Role participantsRole = reworkRole;

java.util.Map<wt.project.Role, wt.fc.collections.WTSet> roleToParticipantMap;

// Try to get participants from the rework activity.

roleToParticipantMap = com.ptc.windchill.pdmlink.change.server.impl.WorkflowProcessHelper.getActivityParticipants(

(wt.change2.VersionableChangeItem) primaryBusinessObject, "Rework Change Notice Task");

// If the map is null then the rework activity has not executed yet.

if (roleToParticipantMap == null) {

participantsRole = wt.project.Role.toRole("ASSIGNEE");

// Try to get participants from the original activity.

roleToParticipantMap = com.ptc.windchill.pdmlink.change.server.impl.WorkflowProcessHelper.getActivityParticipants(

(wt.change2.VersionableChangeItem) primaryBusinessObject, "Complete Change Notice Task");

}

// Assign participants from original or rework activity to the rework role on the change team.

// By doing so only those participants will get the next rework tasks.

wt.fc.collections.WTSet participants = roleToParticipantMap.get(participantsRole);

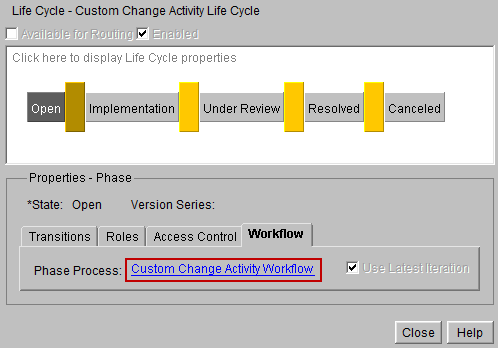
com.ptc.windchill.pdmlink.change.server.impl.WorkflowProcessHelper.setChangeItemParticipants(

(wt.change2.VersionableChangeItem) primaryBusinessObject, reworkRole, participants);

1. When Java code is complete select the “Check Syntax” button to check for errors. Be sure to correct all errors before using the custom workflow template.
2. Save the changes made to the “Custom Change Activity Workflow” template and check it in.

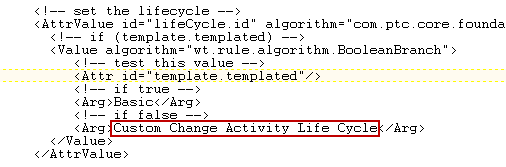
## Customize Change Activity Life Cycle

Create a custom “Change Activity Life Cycle” template using the Life Cycle Template Administration utility. Specify the custom workflow template created in the previous section:



## Customize Change Activity Object Initialization Rule

Create a custom “Change Activity” object initialization rule (OIR) using the template using the Object Initialization Rules Administration utility. Specify the custom life cycle template created in the previous section:



# Java Helper Methods

Several Java helper methods are provided to simplify the code for automatically making rework assignments (see code Java above). These methods are located in class:

com.ptc.windchill.pdmlink.change.server.impl.WorkflowProcessHelper

## Helper Method getActivityParticipants

/\*\*

\* Get participants of a completed work activity for a change item. If the work activity was

\* executed multiple times (e.g., a rework activity was run several times), then only the

\* participants from the latest completed execution are returned.

\*

\* <BR><BR><B>Supported API: </B>true

\*

\* **@param** changeItem Work flow primary business object.

\* **@param** activityName Work flow activity name.

\*

\* **@return** Map of activity roles to the participants for that role, or null if an

\* activity with the specified name has not been executed for the change item.

\*

\* **@throws** WTException

\*/

**public** **static** Map<Role, WTSet> getActivityParticipants(VersionableChangeItem changeItem,

String activityName) **throws** WTException

## Helper Method getChangeItemParticipants

/\*\*

\* Get all participants of a change item team.

\*

\* <BR><BR><B>Supported API: </B>true

\*

\* **@param** changeItem Change item object.

\*

\* **@return** Map of change item roles to the participants for that role.

\*

\* **@throws** WTException

\*/

**public** **static** Map<Role,WTSet> getChangeItemParticipants(VersionableChangeItem changeItem)

**throws** WTException {

## Helper Method setChangeItemParticipants

/\*\*

\* Set the participants for a role of a change item team. The old participants of the

\* role, if any, will be replaced with the new participants.

\*

\* <BR><BR><B>Supported API: </B>true

\*

\* **@param** changeItem Work flow primary business object.

\* **@param** role Change item team role to be set.

\*

\* **@param** participants Collection of new participants for the role. Pass an empty set to remove

\* all participants from the role. Pass null to remove the entire role.

\*

\* **@throws** WTException

\*/

**public** **static** **void** setChangeItemParticipants(VersionableChangeItem changeItem, Role role,

WTCollection participants) **throws** WTException {