Demo on Building Workflow Using Custom Business Object, Events and Methods

SAP

Applies to:

SAP R/3 version 4.6c and onwards. For more information, visit the [Business Process Modeling homepage](#).

Summary

The article describes the procedure to create a workflow with custom business object, having custom methods and custom events.

Author: Saba Sayed

Company: Larsen & Toubro Infotech Limited

Created on: 07th August 2009

Author Bio

Saba Sayed is SAP certified NetWeaver ABAP Consultant, working in Larsen & Toubro Infotech Limited. She has more than three years of SAP experience and has worked extensively in ABAP, OO ABAP, Workflow, ALE / IDoc and SAP SRM.
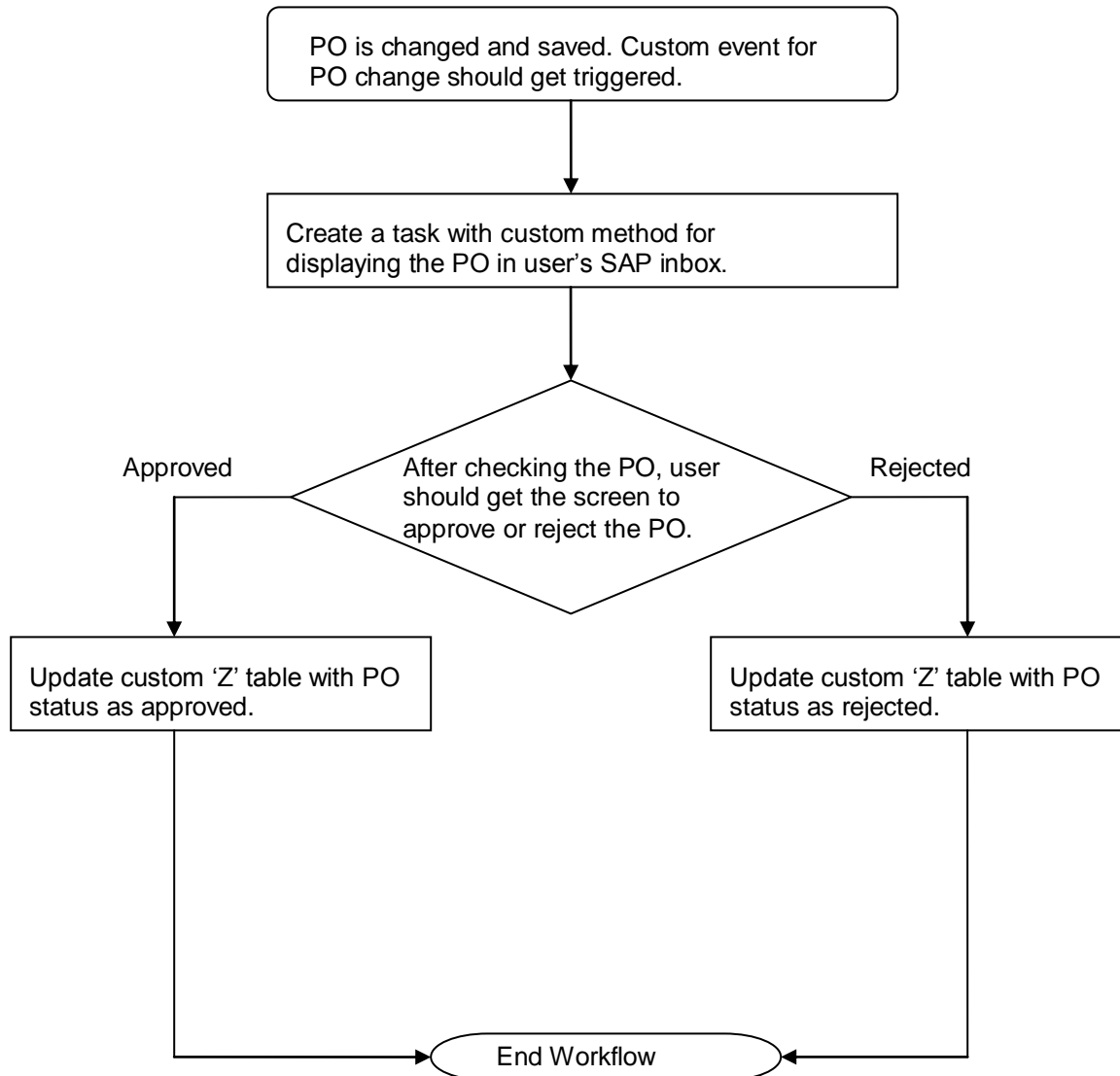
Table of Contents

Scenario

Create a workflow having custom Business object referencing standard Business Object for Purchase Order (BUS2012). In this BO, create a custom method for displaying PO (PODisplayNew) similar to standard method 'Display' and a custom event (POChanged) that should be raised whenever you save the PO.

When the user approves/rejects the PO, a background task should run updating the custom ['Z'] table with status of PO as approved/rejected respectively.

1. Flowchart

```
┌──────────────────────────────────────┐
│ PO is changed and saved. Custom event │
│ for PO change should get triggered.    │
└──────────────────────────────────────┘
                  │
                  ▼
┌──────────────────────────────────────┐
│ Create a task with custom method for  │
│ displaying the PO in user's SAP inbox. │
└──────────────────────────────────────┘
                  │
                  ▼
```

Approved   After checking the PO, user   Rejected
                         should get the screen to
                         approve or reject the PO.

```
┌──────────────────────────────┐          ┌──────────────────────────────┐
│ Update custom 'Z' table with PO │       │ Update custom 'Z' table with PO │
│ status as approved.            │        │ status as rejected.            │
└──────────────────────────────┘          └──────────────────────────────┘
```
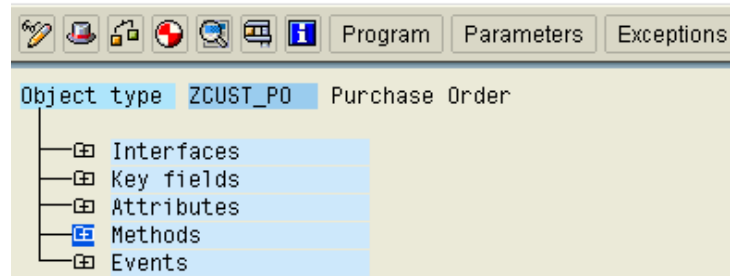
End Workflow

2.   Step-by-step procedure

2.1   Create business object

Create a custom business object referencing the standard business object 'BUS2012' and custom event for changed PO.

    a. Copy the standard business object and create a new business object OR you can create business object by creating standard business object as supertype and new one as subtype.
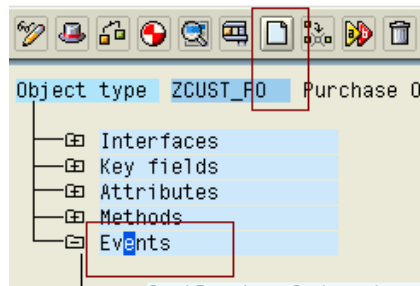

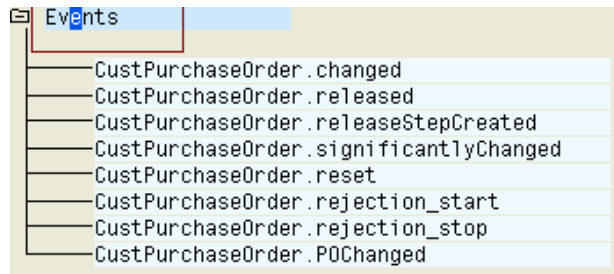Here I have created new BO that is a copy of standard BO 'BUS2012'.

2.2   Create Event

    a. Create new event under the 'Events' node by selecting 'Events' and clicking on 'Create' button.


    b. Enter the required details for the event and click the green arrow.

   c. Check your event will be created under 'Events' node.

d. Change the status of Event as implemented and also change the status of business object as implemented 'Edit→ Change release status → Object type → To Implemented'. Then generate the business object by clicking on red and white circle 'Generate' on application toolbar. Now your business object will be available for use.
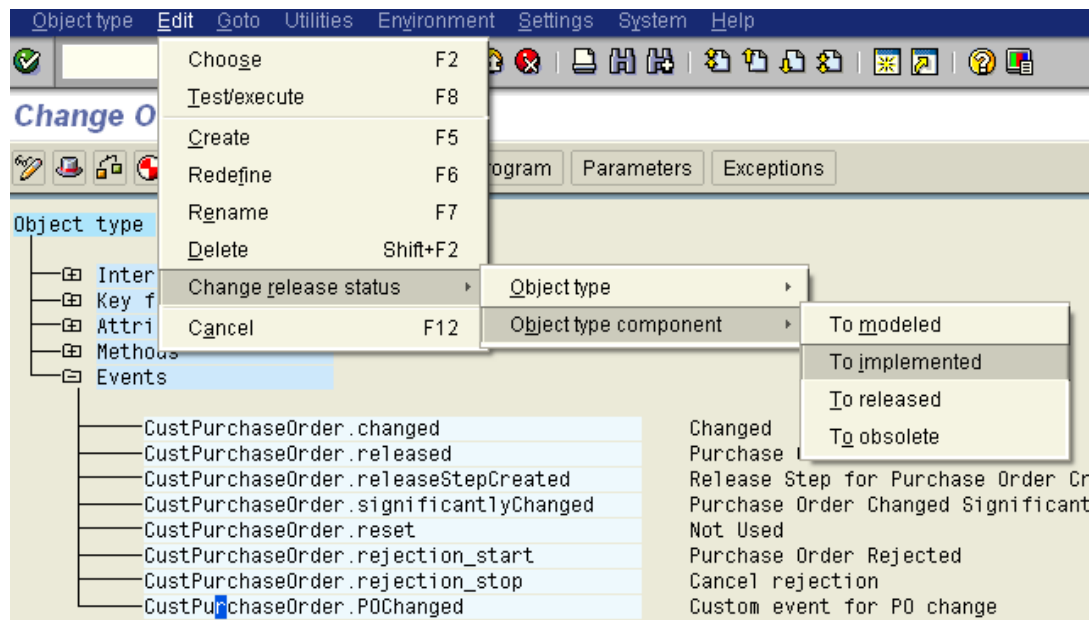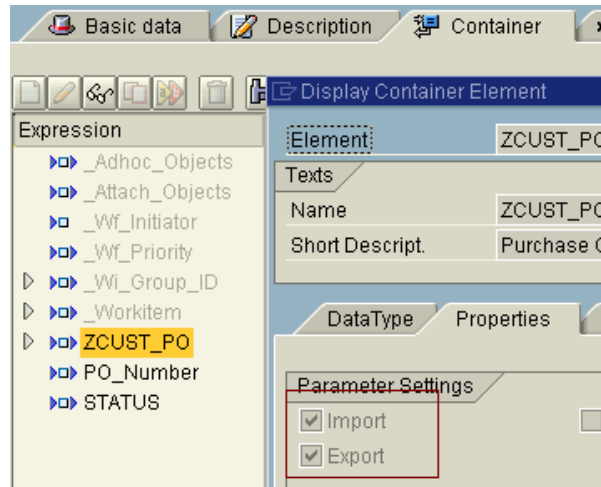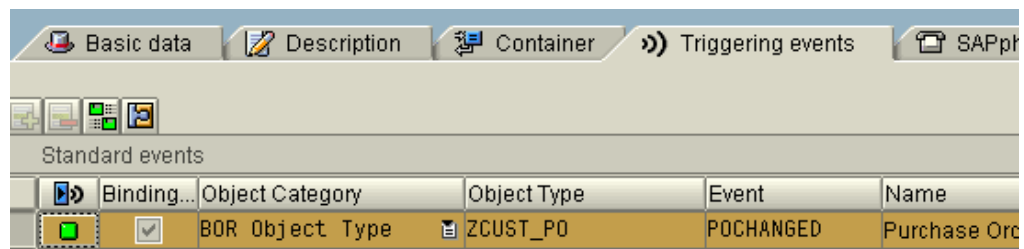

2.3   Trigger Event

a. To trigger your event when the PO is changed and saved,
    i. Create a workflow template from transaction 'PFTC_INS'. Enter 'Basic Data'.
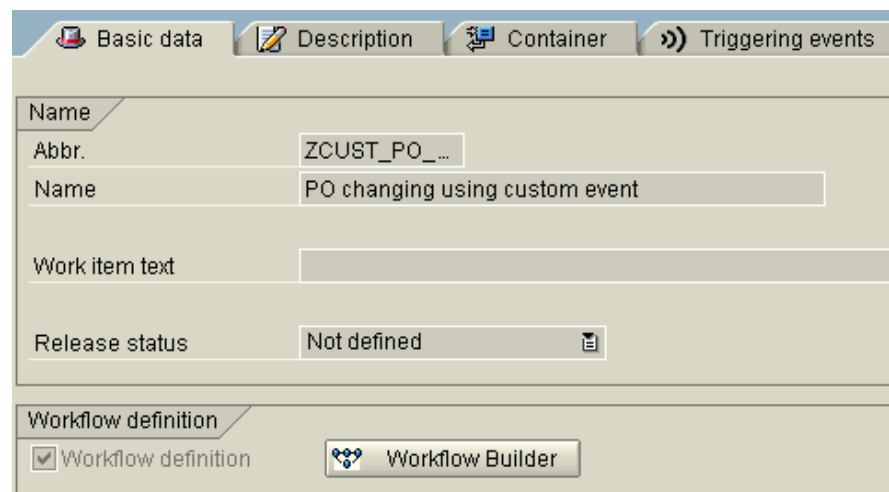
Go to 'Container' tab and enter the elements required for your workflow. Note that in properties select atleast one parameter.

Finally in the 'Triggering Events' tab, enter the event details, do the binding and activate the event line item.

This completes your event creation process in workflow. Now for raising and triggering the event follow step ii.

Check in 'Workflow builder' [click on 'Workflow builder' button in 'Basic Data' tab or go to 'SWDD' transaction and enter the workflow template id] you will see the event is created.

ii. Find out the user exit that is available for PO in case of 'Save' event. Please note that <u>only when the PO is saved the event should get triggered</u>. In this case the exit is EXIT_SAPMM06E_013 that is invoked prior to "commit work". Create and raise the event for POChanged in the above mentioned exit. [Include ZXM06U44]

```
*** Code to raise custom event...done by Saba...pl do not delete ***
*Data declaration
DATA: objkey TYPE swr_struct-object_key.
*Internal tables and workarea declaration
DATA: lt_msglines TYPE STANDARD TABLE OF swr_messag,
      lt_msgstruc TYPE STANDARD TABLE OF swr_mstruc,
      wa_msglines TYPE swr_messag,
      wa_msgstruc TYPE swr_mstruc.

* Assigning PO value to object key
objkey = i_ekko-ebeln.

CALL FUNCTION 'SAP_WAPI_CREATE_EVENT'
  EXPORTING
    object_type = 'ZCUST_PO'
    object_key  = objkey
    event       = 'POChanged'
*    COMMIT_WORK             = 'X'
*    EVENT_LANGUAGE          = SY-LANGU
*    LANGUAGE                = SY-LANGU
*    USER                    = SY-UNAME
* IMPORTING
*    RETURN_CODE             =
*    EVENT_ID                =
  TABLES
*    INPUT_CONTAINER         =
    message_lines           = lt_msglines.
*    MESSAGE_STRUCT          = MSG_STRUC.

IF sy-subrc = 0.
  COMMIT WORK.
ELSE.
  LOOP AT lt_msglines INTO wa_msglines.
    MESSAGE e004(zex) WITH wa_msglines-line.
  ENDLOOP.
ENDIF.

* Clear and refresh internal tables and workarea
clear : lt_msglines, wa_msglines.
refresh lt_msglines.

*** End of Code....by Saba **********
```

      b. Now whenever a PO is changed and saved, the above custom event will be triggered.

2.4   Create Method

a. Creating custom method to display the PO.
b. Create method in the similar way as that of event creation i.e. select 'Methods' and click on create. It will ask you 'Create with function module as template?' then select 'No' as we want to create a 'Dialog' and 'Synchronous' method for displaying the PO. Change the status of method as implemented.

c. Select the method and click on 'Program' button. It will ask you if you want to generate a template automatically for the missing section. Click on 'Yes'. It will navigate you to program with begin and end of method block. Write the code for the method in this block.

```
BEGIN_METHOD PODISPLAYNEW CHANGING CONTAINER.

CALL FUNCTION 'ME_DISPLAY_PURCHASE_DOCUMENT'
  EXPORTING
    i_ebeln = object-key-purchaseorder
    i_enjoy = 'X'
    i_bstyp = 'F'
  EXCEPTIONS
    OTHERS  = 1.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE 'S' NUMBER sy-msgno
          WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.
END_METHOD
```

d. Once this is done click on 'Generate' button. Now this method could be used in the workflow task.

2.5   Create Step

2.5.1   Activity Step-Display PO

a. Go to workflow builder and double click to add a new step. Select the step as 'Activity' as we just want to display the PO whenever the event is triggered.

b. Create a new task in the step and enter the business object details in it. Save it.

Object method		
Object Category	BOR Object Type	
Object Type	ZCUST_PO	Purchase Or
Method	PODISPLAYNEW	

c. Do the binding. Once binding is done you can see green color.

Task		TS99900586	Display ne
Step name		Display new PO	
		Binding (Exists)	

d. Assign the agents as workflow initiator or any user whose inbox this display PO activity should go.

Agents	
Expression	& WF INITIATOR&

e. In 'Task Properties', set the attribute of 'Agent Assignment' as 'General Task' by clicking red button next to it.

Task Properties	
🔴 🟩	Agent Assignment
◇	Background Processing
🟩	Task Complete

Name	ID	
Display new PO	TS 99900586	General Task

2.5.2 User Decision Step - Provide Approve or Reject Option

User can approve or reject the PO.

a. Create 'User decision' step with title 'Kindly Approve or Reject the Purchase Order'. Assign agent who will approve or reject the PO. In this case, I have entered as workflow initiator. Enter the 'Decision Texts' and its 'Outcome Names'. Decision texts will appear as buttons on the screen where agents can approve or reject the PO while the outcome names will appear as the step text in the workflow logs.

b. After creation of above step, you will find 2 options one as approve and other as reject.

2.5.3 Steps to Approve PO

a. Double click to create a step for approve option. Select this step as 'Activity' *[Update custom table with approve/reject]. Note that we need to create a background step that will update the custom 'Z' table with status as either approved/rejected. For this we will have to first create a new method 'UpdateTable' under 'Methods' node in 'ZCUST_PO' business object. In order to create task as background, note that the method should not be a dialog method i.e. uncheck 'Dialog' in method properties.

b. Create a 'Z' database table with fields' mandt, po number, date, time, status and description.

c. To get the value of approval or rejection from User Decision (UD) step, you will have to create additional step between UD step and approval activity step. This step could be created by double clicking [or right click and create] *'PO Approved' i.e. the outcome name as highlighted below. Select this step as 'ContainerOperation' *[zapp_po]. After creation of this step your workflow will look like as given below:

*[P.S.: * indicates check the below screenshot for step information]*
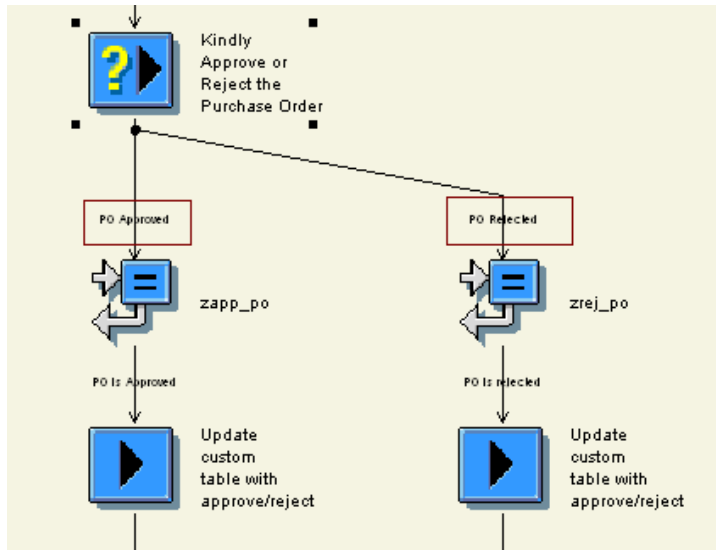
d. To capture the status of PO create container element status. In properties, check import and export parameter.

e. Now in the 'ContainerOperation' step for approval, enter the following details:

2.5.4   Steps to Reject PO

Follow the same steps as of **step 5.3**. The only difference would be the 'Expression' value, which will be 'R' instead of 'A'.

2.6   Create Task

a. Now for the main background step i.e. for updating the table, create a new task in this step with following details:

Object method	
Object Category	BOR Object Type
Object Type	ZCUST_PO Purchase (
Method	UPDATETABLE

Execution	
☑ Background processing	☐ Executable w
☐ Confirm end of processing	

Select execution method as background as we want the update should happen in background.

In the task container, create container as 'Status' so that you can bind this task container element with the container operation step's Status value.

| 🖼 Container | |

| ▶□▶ STATUS | Status: Approved or Rejected | < Not Set > |

| 🔴 🖼 Binding (Exists) |

Do the binding for the task d background tasks) linking status of workflow container with the status of task container. (Note that datatype and length of both the container elements should be the same otherwise binding will not happen.)

Binding Workflow -> Step 'Update custom table with approve/reject'

Workflow			Step 'Update custom t
&ZCUST_PO&	⊡	➡	&_WI_OBJECT_ID&
&STATUS&		➡	&STATUS&
		➡	

Binding Workflow <- Step 'Update custom table with approve/reject'

Workflow			Step 'Update custom t
&ZCUST_PO&	⊡	⬅	&_WI_OBJECT_ID&
&STATUS&		⬅	&STATUS&

b. Do the same for rejection activity also. Take the same task name as created for approved activity. No need to create a new task as the properties of both the tasks will be same.

2.7 Code for Method

a. Finally we need to write code for 'UpdateTable' method. It is said that the table should be updated on the value received from UD step.

```
BEGIN_METHOD UPDATETABLE CHANGING CONTAINER.
DATA: STATUS TYPE ZAPP_REJ-STATUS.
SWC_GET_ELEMENT CONTAINER 'STATUS' STATUS.
CALL FUNCTION 'ZPO_UPDATE_TABLE'
  EXPORTING
    ebeln               = object-key-purchaseorder
    status              = STATUS.
* IMPORTING
*   IT_ZAPP_REJ         =
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE 'S' NUMBER sy-msgno
          WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.
SWC_SET_ELEMENT CONTAINER 'STATUS' STATUS.
END_METHOD.
```

Write the above code in business object program. Above we are declaring a variable 'Status' to accept approve/reject i.e. A or R value. Next get the value of 'Status' [container element declared in task]. This will contain the value that will be obtained from 'ContainerOperation' step from the workflow container element 'Status' [as we have done binding for task container and workflow container].

[In the command, "SWC_GET_ELEMENT CONTAINER 'STATUS' STATUS", container is the container for the element, first 'status' is the element name and second status is the field/variable that you have declared in the method.]

It could be seen that in the method, we are calling F.M 'ZPO _UPDATE_TABLE'. It will have the following code in it:

```
* Initialization
  w_date = sy-datum.
  w_time = sy-uzeit.

* Assign PO and status value based on user's approval or rejection of PO
  w_ebeln = ebeln.
  w_status = status.

* Update custom table based on approval or rejection of PO
  IF w_status = 'A'.                          " PO Approved in workflow
    wa_apprej-ebeln  = w_ebeln.
    wa_apprej-ardate = w_date.
    wa_apprej-artime = w_time.
    wa_apprej-status = w_status.
    wa_apprej-descr  = text-001.
    APPEND wa_apprej TO gt_apprej.
    CLEAR wa_apprej.
  ELSEIF w_status = 'R'.                       " PO Rejected in workflow
    wa_apprej-ebeln  = w_ebeln.
    wa_apprej-ardate = w_date.
    wa_apprej-artime = w_time.
    wa_apprej-status = w_status.
    wa_apprej-descr  = text-002.
    APPEND wa_apprej TO gt_apprej.
    CLEAR wa_apprej.
  ENDIF.

* Insert into database table.
  INSERT zapp_rej FROM TABLE gt_apprej.

* Clear variables
  CLEAR : w_ebeln, w_date, w_time, w_status.

ENDFUNCTION.
```

2.8 Workflow Completed

Now your workflow is completed. Save it and activate it.

2.9 Test Workflow

Test the workflow. Go to transaction ME22N or ME22. Enter the PO number '4500016119' and change the quantity and save it. As soon as the PO is saved, a workitem will appear in your inbox (as we have kept the agent as workflow initiator).
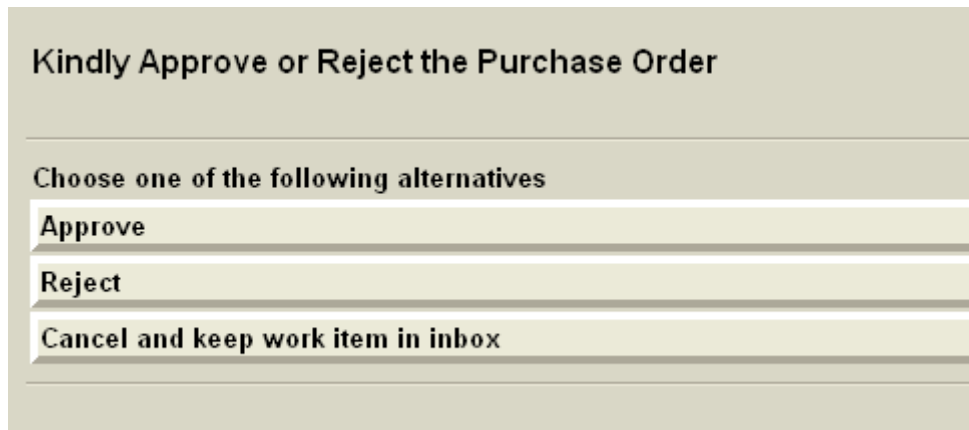
2.10   Business Workspace

a. Go to SBWP to check the workitem. 'Inbox → Workflow' path. You will see the task description on the right hand side.
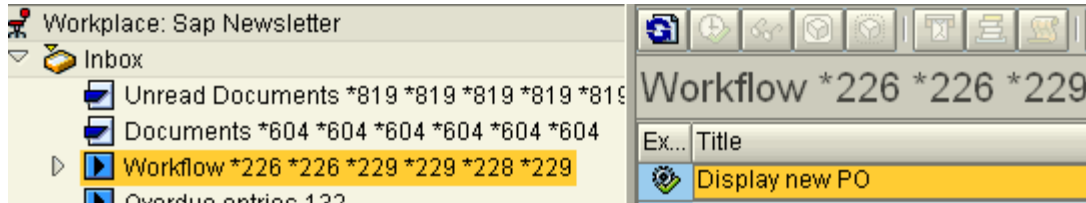
b. Select this workitem and click on execute button next to refresh button on right hand side or just double click 'Display new PO'. It will take you to the PO display screen.
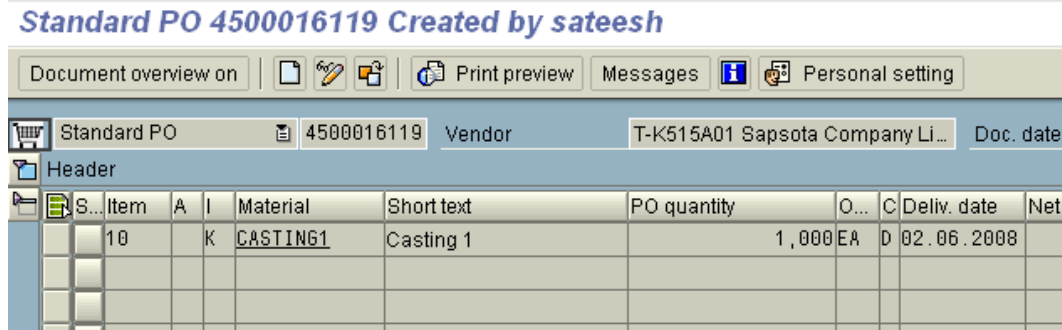
c.   When you come out of this transaction by clicking on any of these buttons, it will call next screen for PO approval or rejection.

Kindly Approve or Reject the Purchase Order

Choose one of the following alternatives

Approve

Reject

Cancel and keep work item in inbox

d.   When you select 'Approve' it will update database table with status as 'A' and if you 'Reject' then status will be 'R'. In this case, I have approved the PO.

Check the table ZAPP_REJ, to see if the values are reflected correctly in the table.

MANDT	EBELN	ARDATE	ARTIME	STATUS	DESCR
800	4500016129	22.09.2008	12:05:02	A	PO is approved

e.   This completes our workflow.

2.11   Workflow Log

You can check the workflow log through transaction 'SWI1'. Enter the task as workflow template ID and execute. You will get the complete overview of the steps, container elements and agents from this log.

2.12   Workflow Transactions

Transaction Code	Description
PFTC	General Task Maintenance
PFTC_DIS	Display Task
PFTC_CHG	Maintain Task
PFTC_INS	Create Task
SWDD	Workflow Definition
SBWP	SAP Inbox
SWO1	Object Builder
SWI1	Selection reports for Workflow

2.13   Purchase Order Transactions

Transaction Code	Description
ME21N/ME21	PO Create
ME22N/ME22	PO Change
ME23N/ME23	PO Display

Related Content

https://wiki.sdn.sap.com/wiki/display/ABAP/SAP+Business+Workflow

https://wiki.sdn.sap.com/wiki/display/ABAP/SAP+Workflow+EVENTS

For more information, visit the Business Process Modeling homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.