
SAPTips: Workflow Troubleshooting and Debugging

Document Prepared By
Thomas Nittmann

Introduction	3
Primary Components	3
Base Line Configuration	3
Event Debugging.....	12
Problem1 Event Linkage:	14
Resolution to Problem 1:	14
Problem 2 Event Parameters:	15
Resolution A to Problem 2:.....	15
Resolution B to Problem 2:.....	23
Additional Event Information:	27
Agent Resolution Debugging	32
Problem 3 Agent Resolution:.....	32
Resolution to Problem 3:	32
Workflow Task (Method) Debugging.....	46
Foreground (Dialog) Tasks.....	46
Background (Synchronous) Tasks	51
Summary of Trouble Shooting Transactions.....	65

Introduction

This article focuses on practical techniques used to assist in the debugging and resolving workflow issues during the development and production support phases of SAP implementations. The SAP Workflow application incorporates the use of several components, i.e. graphical editing tool (workflow builder), some object orientated concepts (use of the Business Object Repository), ABAP, and a hook/trigger mechanism into one of the SAP application modules (FI, CO, MM, etc). As business requirements are mapped into a workflow process definition and prototyping starts, **debugging** becomes an essential tool to expedite the implementation of workflow process.

We will concentrate on using debugging techniques to resolve real world workflow implementation and production support issues. Before we begin let's list some preliminary assumptions and guidelines:

- The workflow engine has been enabled
- We are working with SAP Release 46C. There are significant changes and enhancement in release 6.1/6.2 SAP workflow technology
- The user should be familiar with basic workflow configuration and development, e.g. the workflow type 'TS' is a single step task and type 'WS' is a multi-step task.

Since workflow has many different elements, starting the debugger is not as straight forward as with a stand alone ABAP program. To illustrate the best way to step in and resolve issues by debugging we will start by looking at a **COPY** of the standard SAP workflow template for the overall release of MM purchase requisitions.

The debugging principles applied may be used with any workflow templates (customer defined or standard). A custom workflow template (WS) is being used to illustrate the use of debugging concepts on background tasks. A new custom single step task (TS) has been created for notifying the creator of the purchase requisition release and replaces the DIALOG FOREGROUND TASK that requests the creator manually confirm a work item.

Primary Components

Each workflow consists of several components that are best described by answering the following questions:

- When should the workflow start? The answer to this question assists in determining the EVENT TRIGGERING mechanism that starts the workflow.
- What steps need to occur, i.e. what TASKS (transactions and/or background items need to be performed) need to occur?
- Who performs the tasks, i.e. the AGENT RESOLUTION?

As we proceed we will review problem solving and debugging techniques used in each of the components.

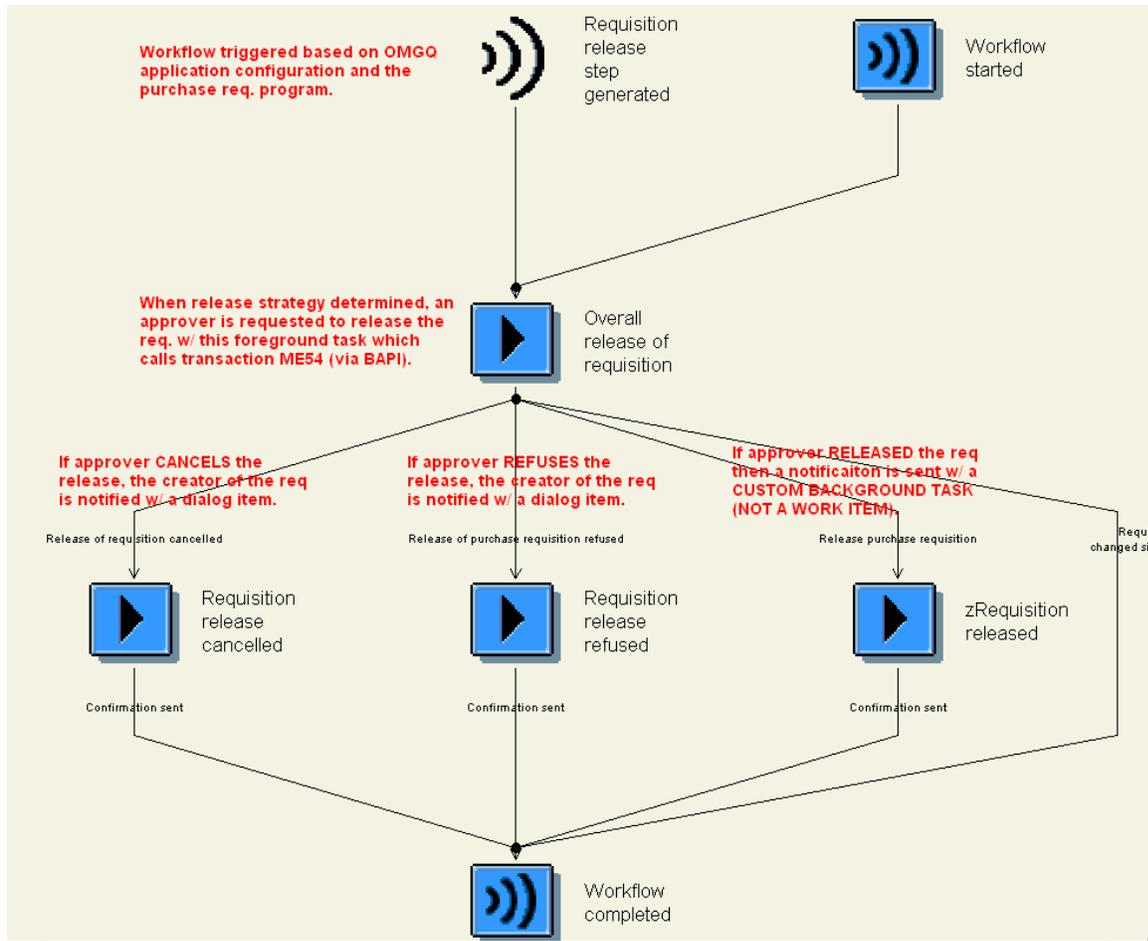
Base Line Configuration

First, let's break down the CUSTOM release workflow into its primary multi-step and single-step tasks.

- Custom Workflow Template **WS99500034** (zwf_req_rel) is the multi-step task for the overall purchase requisition and is a copy of the standard template WS20000077. It contains several single-step (TS) tasks.
- Single-step task **TS20000159** (mm_req_rel_c) is a DIALOG FOREGROUND task for the overall release of the workflow. Method BUS2105-SINGLERELEASE.
- Single-step task **TS20000163** (mm_req_res_c) is a DIALOG FOREGROUND task that requests the creator of the requisition confirm that the Requisition Release was Cancelled. Method BUS2105-INFORELEASERESET.
- Single-step task **TS20000161** (mm_req_rej_c) is a DIALOG FOREGROUND task that requests the creator of the requisition confirm that the Requisition Release was refused. Method BUS2105-INFORELEASEREJECTED.
- Single-step task **TS99500020** (zmm_req_ok) is a CUSTOM BACKGROUND task that replaces the standard task TS20000162 (Requisition Released). The concept to have a background task is to eliminate the additional work that would be required by confirming a FOREGROUND DIALOG work item, i.e. we only want to send a NOTIFICATION and not a WORK ITEM to the creator of the requisition. We enable a NOFTIFICATION with the CUSTOM BACKGROUND task and method ZBUS2105-ZNOTIFYREQRELEASE.

Next, review the requisition release Workflow Builder Process Flow:

Note: to display the delivered SAP Workflow use transaction PFTC > Enter 'Workflow Template' for Task Type > enter '20000077' for task > click on display > click on workflow builder.



The requisition overall release workflow is pre-delivered by SAP and even though we created a custom version of the workflow there is some baseline configuration required to activate it. Many (but not all) pre-defined workflows require some sort of IMG configuration. Let's quickly review the baseline configuration:

*Note: to find the on-line R/3 documentation for workflow navigate to:
 Basis Components > Business Management (BC-BMT) > SAP Business
 Workflow (BC-BMT-WFM) > Reference Documentation > BC – Workflow
 Scenarios in Applications (BC-BMT-WFM) > MM – Materials Management:
 Workflow Scenarios*

1. Use transaction OMEB to activate the Overall Release for the purchase requisition document type.

Note: SAP initially delivered line item release strategies and latter enhanced the application with an overall release. The purchase req. application is built on one table (EBAN) and is NOT separated into header and line item tables like most other applications.

Type	Doc. type descript.	Item...	NR int.as.	No.rng.ext	Field sel.	Cont...	OverRe...	Variar
EC	Purch.requis. B2B	10	01	RQ	NBB		<input type="checkbox"/>	
FO	Framework requis.	10	01	02	FOF		<input type="checkbox"/>	SRV
IN	Purch.requis. I-Comm	10	01	02	NBB		<input type="checkbox"/>	
MV	Model specification	10	01	02	RVB	R	<input type="checkbox"/>	
NB	Purch.requis. Stand.	10	01	02	NBB		<input checked="" type="checkbox"/>	
RV	Outl. agmt. requis.	10	01	02	RVB	R	<input type="checkbox"/>	

2. Classification (area menu transaction CL00) is required to set-up class and characteristic values used to determine the release strategy.

Create Characteristic Values (CT01)

Create Characteristic

Characteristic: FRG_EBAN_GFWRT

Change number:

Valid from: 25.02.2003

Validity:

Basic data | Descriptions | Values | Addnl data | Restrictions

Basic data

Description: Total value of requisition for

Chars group:

Status: Released

AuthGrp:

Format

Data type: Currency format

No. of chars: 15

Decimal places: 2

Currency: USD

Template: _____, -

Value assignment

Single-value

Multiple values

Interval values

Negative vals allowed

Restrictable

Entry required

Create Characteristic

Characteristic: FRG_EBAN_GFWRT

Change number: []

Valid from: 25.02.2003

Validity: []

Reference to table field:

Table name: CEBAN Field name: GFWRT

Document:

Document: []

Document type: []

Document part: []

Version: []

Procedure for value assignment:

- Not ready for input
- No display
- Display allowed values

User entry handling:

- Unformatted entry
- Propose template

Create Characteristic

Characteristic: FRG_EBAN_GFWRT

Change number: []

Valid from: 25.02.2003

Validity: []

Additional values: Other Value Check: []

Allowed Values:

Char. value	D	O
>= 1000,00 USD	<input type="checkbox"/>	<input type="checkbox"/>

Set-up an overall release class (CL01), assign it to class type '032', and Assign Characteristic Value to it.

Class Basic Data

Change Class:

Change language

Class: FRG_EBAN_ALL

Class type: 032 Release strategy

Valid from: 25.02.2003

Validity

Basic data | Keywords | Char. | Texts

Basic data

Description: Requisition Overall release strategy

Status: Released

Class group:

Organizational area: Local class

Valid from: 25.02.2003 Valid to: 31.12.9999

Same classification

Do not check
 Warning message
 Check with error

Authorizations

Class maintenance:
 Classification:
 Find object:

Administrative data

Assignments

Created by: TNITTMANN Created on: 25.02.2003
 Changed by: TNITTMANN Changed on: 25.02.2003

Class Assignment to Characteristic Values

Change Class:

Change language

Class: FRG_EBAN_ALL

Class type: 032 Release strategy

Valid from: 25.02.2003

Validity

Basic data | Keywords | Char. | Texts

Characteristic	Description	Data...	N...	D...	Unit	R..	Org.areas	DIN ...	O..I...
FRG_EBAN_GFWRT	total value of requisition	CURR	15	2	USD				

3. Transaction OMGQ is required to activate the event trigger mechanism from the MM requisition application.

Release Group:

Change View "Release Groups": Overview

New entries

Rel. group	Rel.object	OverReqRel	Class	Description
01	1	<input checked="" type="checkbox"/>	FRG_EBAN_ALL	reqn release

Release code:

Change View "Release Codes": Overview

New entries

Grp	Code	Workflow	Description
01	EX	1	executive officer

Release Indicators:

Change View "Release Indicator": Overview

New entries

Release indicator	Description
1	Request for quotation
2	RFQ/purchase order
3	RFQ/PO no change of date
4	RFQ/PO no changes
A	Fixed RFQ
B	Fixed RFQ/purchase order
S	Blocked

Release Strategy

Change View "Release Strategies": Overview

Grp	Strat	Description
01	AF	Release/BoD US
01	I2	Internet
01	IN	Internet TD/KY
01	RT	Rel. Training LO520
01	ST	SAPTips Workflow
01	TF	Release/BoD GE
01	ZZ	Rel. Training LO520

Change View "Release Strategies": Details

Release group: 01 reqn release group

Rel. strategy: ST SAPTips Workflow

Release codes:

- EX executive officer
-
-
-
-
-

Buttons: Release prerequisites, Release statuses, Classification, Release simulation

Release Statuses

EX

Release indicator:

- S Blocked
- 2 RFQ/purchase order

Buttons: Continue, Cancel



Workflow

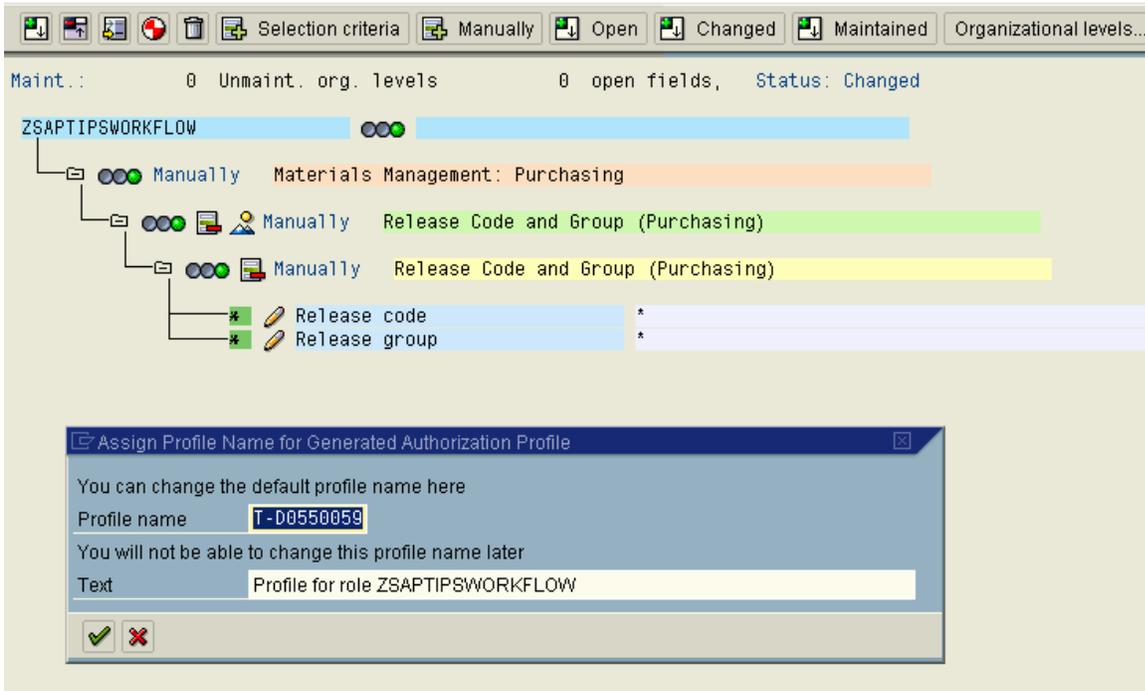
Change View "Assignment of Role to Release Code": Overview

Grp	Co...	Description	Plnt	Ob	Agent ID
01	EX	executive officer		US	TNITTMANN
01	EX	executive officer	1000	US	TNITTMANN
01	EX	executive officer	2400	US	WF-MM-4
01	EX	executive officer	2500	US	WF-MM-4
01	EX	executive officer	3000	US	WF-MM-4
01	EX	executive officer	3800	US	WF-MM-4
01	EX	executive officer	5000	US	WF-MM-4
01	EX	executive officer	5100	US	WF-MM-4

Note: we are currently using the default agent resolution provided by SAP release strategy configuration (OMGQ). We will discuss the use of plant in the agent resolution process below.

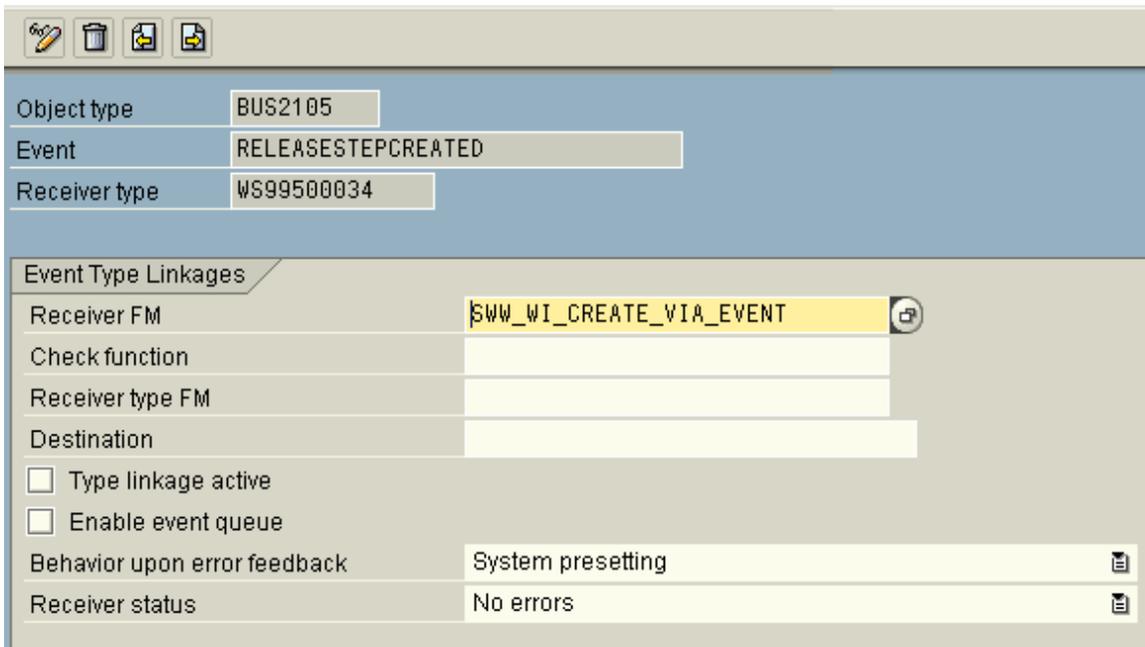
4. Generate Class Statistics via transaction CLST (class type '032'). This optimizes the SAP search process for all characteristics created for the classification.
5. Any user that releases requisitions must be assigned 'M_EINK_FRGAL' authorization (PFCG/SU01)

Change role: Authorizations



6. Lastly, make sure the event linkage is set-up via SWE2.

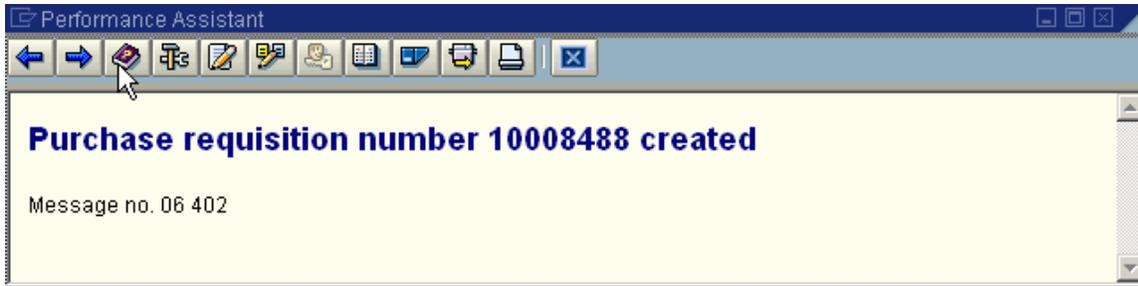
New Entries: Details of Added Entries



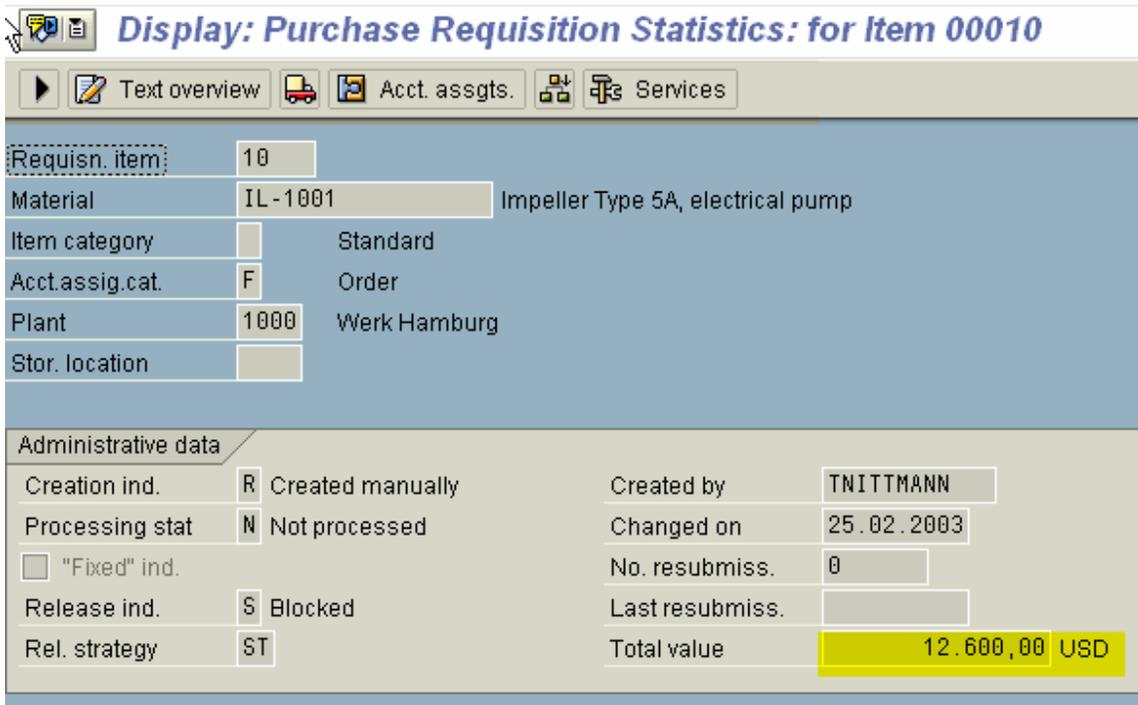
Event Debugging

There was a significant amount of work required to set-up a very simple example of the release workflow for purchase requisitions; however, we are now ready to go! Workflows are primarily triggered based on

events generated by applications. Once the base line configuration is complete we would expect the purchase requisition application to trigger events that will start workflow when we create purchase requisitions that meet our criteria, e.g. any purchase requisition w/ document type 'NB' and a total value over USD1000.00 (based on our OMGQ configuration). Let's create the requisition. The requisition release workflow uses trigger events coded in the original main MM purchasing program <insert program name>. Most often, issues arise with what the parameters that the developer expects to be passed into the workflow container from the event (i.e. the event/workflow template binding condition). To expedite resolution to event/workflow binding issues we will review some helpful debugging and troubleshooting tips:



The total value of requisition '10008488' is USD12,600.00 so it requires the release strategy we set-up in OMGQ. Great, so we should have a workflow.



Problem1 Event Linkage:

We go to transaction 'SWI1' and notice no workflows exist at all. Let's utilize some standard SAP utility transactions before we start going through the purchase requisition program in debug mode.

- turn on the event log with transaction SWELS
- re-create another purchase requisition by copying the one we just created
- display the event trace w/ transaction SWEL

Display Event Trace

Object type	Event	Current date	Time	Receiver type	Info...	Receiver FM / action
	Trace OFF	25.02.2003	13:58:41	TNITTMANN		
	Trace ON	25.02.2003	13:58:42	TNITTMANN		
BUS2105	RELEASESTEP_CREATED	25.02.2003	14:08:45		Info...	No receiver entered
BUS2105	RELEASESTEP_CREATED	25.02.2003	14:22:18		Info...	No receiver entered

Hint: In former SAP releases a workflow development area menu was available to easily access the most common WF transactions. If you are more comfortable w/ the older area menu escape from the 'Easy Menu' by entering '/n' in the transaction window and then enter transaction 'SWLD' to get to the old area menu.

Resolution to Problem 1:

The event trace shows no event receiver was found. This error is typical if the event linkage is not set-up or not active in transaction SWE2. By reviewing SWE2 we notice that the type linkage was not set to active. To resolve the issue activate the event linkage and be sure to save the entries. To continue the exercise we can either create a new requisition to ensure workflow starts or manually start workflow for the existing requisitions.

Tip: Ideally, in a production environment if workflow didn't start when it should have we would want to manually start it. You can manually create the workflow by either using transaction SWUE (create event) or SWUS (start workflow). It's best to use transaction SWI1 and find a similar workflow that has been generated to get the correct syntax and values for the container elements.

New Entries: Details of Added Entries

Object type	BUS2105
Event	RELEASESTEP_CREATED
Receiver type	WS99500034
Event Type Linkages	
Receiver FM	SWW_WI_CREATE_VIA_EVENT
Check function	
Receiver type FM	
Destination	
<input checked="" type="checkbox"/> Type linkage active	
<input type="checkbox"/> Enable event queue	
Behavior upon error feedback	System presetting 
Receiver status	No errors 

Other items to review if the issue was not resolved:

- SWU3: Check the Workflow Engine – make sure all items are configured properly (all lights are green).
- SWUB (part of SWU3): Validate the Workflow RFC destination configured completely. Synchronize the passwords and test the RFC connection.
- SM58: workflow is based on tRFCs. tRFC errors are logged and can be reviewed using transaction SM58.
- SM21: Check the system log for workflow specific errors.
- ST22: Check for workflow specific short dumps.

Problem 2 Event Parameters:

Often, customers identify additional event parameters that are required in the workflow that are not being passed from the event trigger. In addition, you may expect values that you are not seeing in the workflow so you want to review the code to determine what parameters are actually getting passed from the application to the workflow.

In this specific example SAP provided the 'hooks' (trigger mechanism) to start the workflow directly in the MM purchase requisition application. Workflow starts because we configured the OMGQ transaction to activate workflow – the MM purchase requisition program checks the OMGQ to determine if WF should be started. Several options exist to view the event parameters being passed:

Resolution A to Problem 2:

Debug the MM Purchase Requisition application and look for the Event Create function modules. Events are created using several possible function modules: SWE_EVENT_CREATE, SWE_EVENT_CREATE_IN_UPD_TASK, and SWE_EVENT_CREATE_FOR_UPDATE_TASK. Before we begin with this approach it IS IMPORTANT TO NOTE that this particular application calls the workflow routines from a function module 'in update task', i.e. we can not simply start debug and force a break-point at 'SWE_EVENT_CREATE'. This option is the most work intensive but we will review it to illustrate the process. We will proceed by using transaction ME51 to create a new purchase requisition (w/ similar characteristics as before so the release strategy is evoked). Before saving the requisition we switch the debug mode on by entering '/h' in the transaction window.

The screenshot shows the SAP ME51 transaction window titled "Create: Purchase Requisition: Item 00010". The menu bar includes "Purchase requisition", "Edit", "Goto", "Environment", "System", and "Help". The command field contains "/h". Below the menu bar are several icons for navigation and actions. The main area is divided into several sections:

Req. item	10	Item cat.		AcctAssCat	F
Material	IL-1001	Matl group	011	Plant	1000
Short text	Impeller Type 5A, electrical pump			Stor. loc.	

Qty. and date

Quantity	1	PC	Deliv.date	D 09.12.1994
----------	---	----	------------	--------------

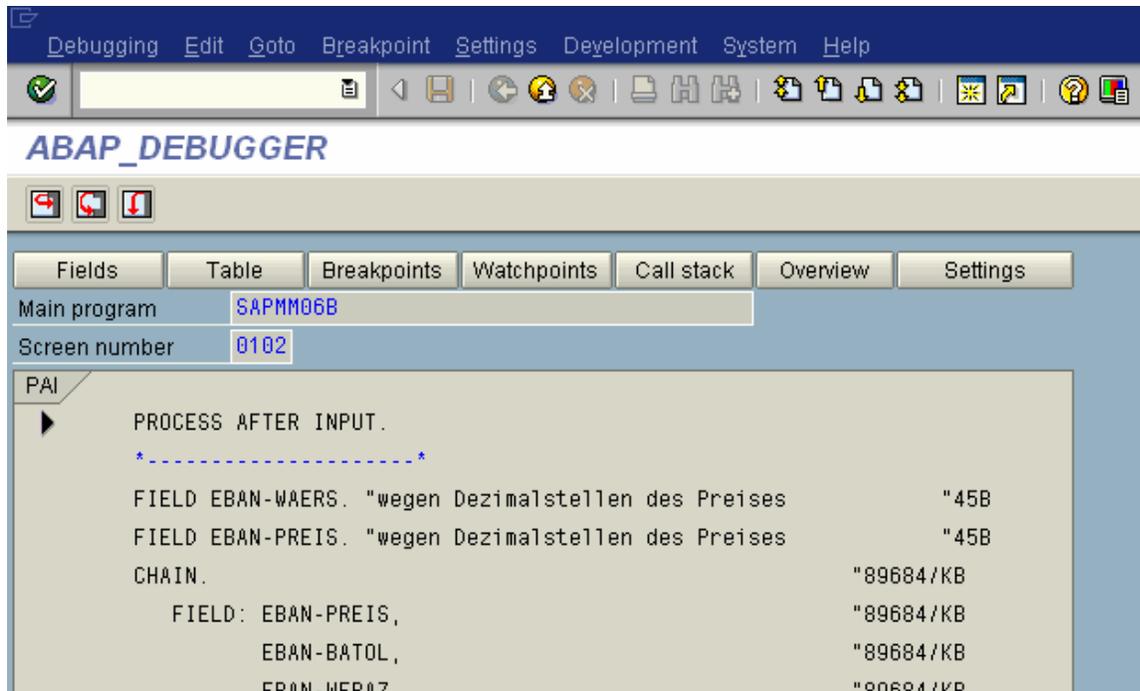
MRP data

Requisnr.	nittmann	Purch. grp	007	Req. date	26.02.2003	Resubmis.	
TrackingNo		MRP ctrlr		Release dt	08.12.1994	GR pr.time	
Rev. level		<input type="checkbox"/> Fixed					

Valuation control

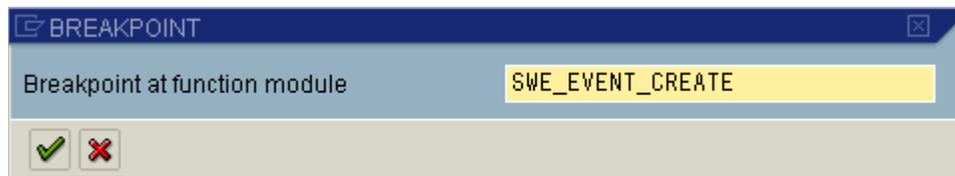
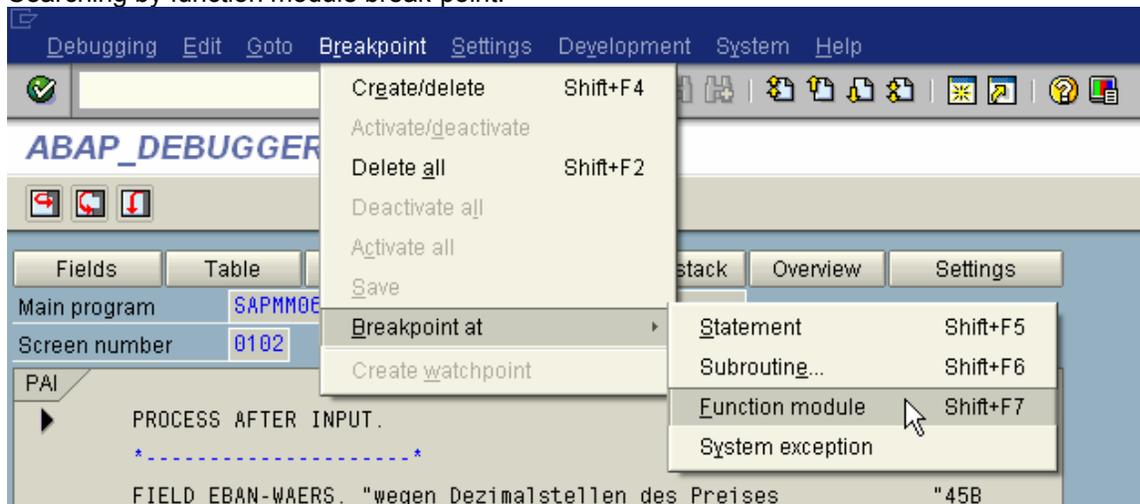
Val. price	1.050,00	USD	/ 1	PC	<input checked="" type="checkbox"/> GR	<input checked="" type="checkbox"/> IR
					<input type="checkbox"/> GR non-val	

Now, we can save the requisition and enter the primary MM program for purchase requisitions.



Next, the fun part begins. Initially, I would suggest try setting a break-point at 'SWE_EVENT_CREATE' and/or go to transaction SE38 or SE80 and enter program SAPMM06B and search for 'SWE_EVENT_CREATE', 'EVENT', 'RELEASE', and/or 'Workflow' to determine where the workflow event is triggered.

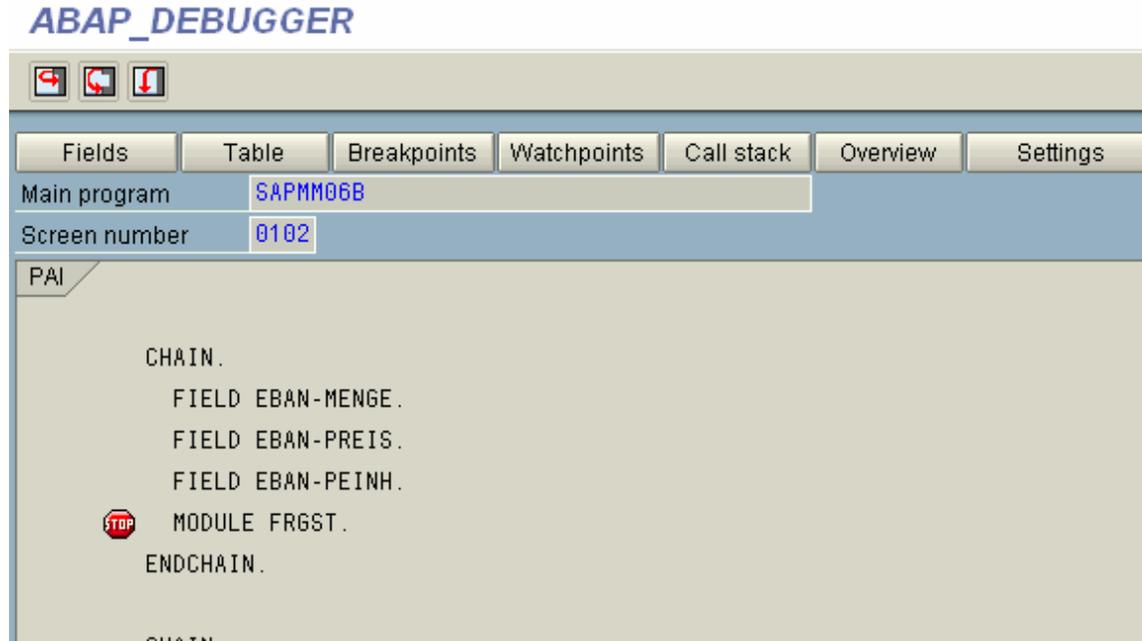
Searching by function module break-point.



By performing this once, we notice that the break point is not reached. Another purchase requisition must be created and we enter debug mode again by entering '/h' in the transaction window.

Hint: Most of the SAP code has its roots from the German language. The abbreviation 'FRG' seen throughout the code (and class system) is short for 'Freigabe Strategie' or 'Release Strategy'. To find out where workflow is triggered search the main program 'SAPMM06B' for the module that begins with 'FRG'.

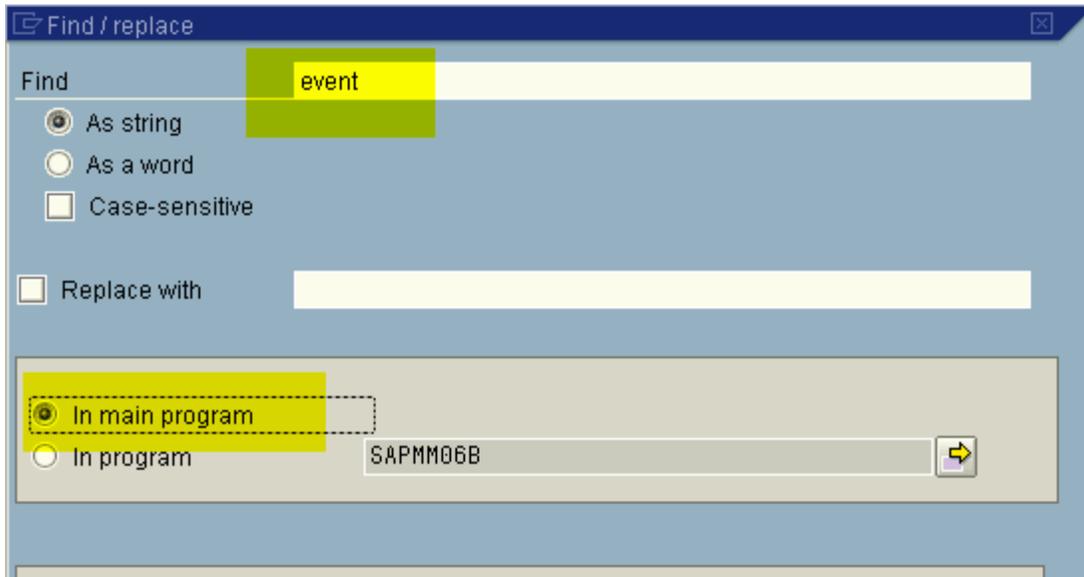
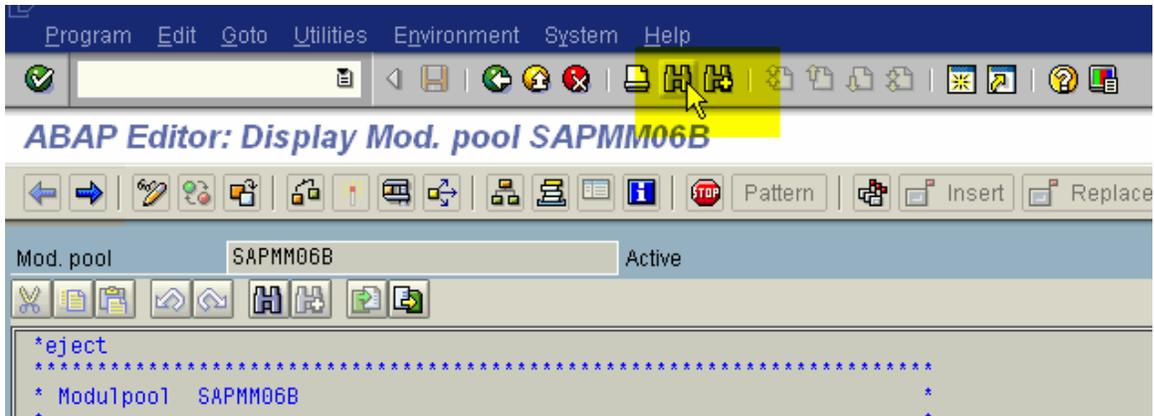
In debug mode we page down and review all the modules and pinpoint 'MODULE FRGST'. Place a break-point here and then continue to the break-point and then step into the module.



The FRGST module is only used to determine the release strategy. Workflow is not started in this module so we must continue.

Note: the function modules ME_REL_STRATEGY_REQUISITION and ME_REL_GENERAL_STRATEGY_EBAN used in module FRGST can be used to troubleshoot potential issues w/ the release strategy determination for purchase requisitions.

Page through the code and try to determine where workflow is triggered. Let's search the primary purchase requisition program 'SAPMM06B' using search criteria 'event'.



Global Search in Programs

Program	Found locs/short description
<input type="checkbox"/> MM_MESSAGES_MAC	141 * Events *
<input type="checkbox"/> MM06B10F_FRGST	47 CALL FUNCTION 'ME_REL_STRATEGY_REQUISITION' EXPORTING I_EBAN_NEW = EBAN * I_EBAN_OLD = I_EBKN_NEW = EBKN * I_EBKN_OLD = I_XCLASS = 'X' I_EVENT = 1 * I_CHANGE_OK = IMPORTING E_EBAN = EBAN * E_RESET = EXCEPTIONS OTHERS = 0. 89 CALL FUNCTION 'ME_REL_STRATEGY_REQUISITION' EXPORTING I_EBAN_NEW = EBAN I_EBAN_OLD = EBAN I_EBKN_NEW = EBKN I_EBKN_OLD = *EBKN I_XCLASS = 'X' I_EVENT = 1 * i_change_ok = IMPORTING E_EBAN = EBAN E_RESET = HRESET EXCEPTIONS OTHERS = 0.
<input type="checkbox"/> MM06B08B_BUCHEN	197 call function 'RWIN_CHECK' exporting gjahr = h-gjahr process = 'BANF ' event = '0INUMBER' tables tkomp = trwin exceptions others = 0. 426 call function 'ME_REL_EVENT_EBAN' in update task exporting i_call_updkz = call_updkz i_wfbn = wfbn i_wfpos = wfpos i_frgco = rm06b-frgab i_ernam = wfern tables t_eban_new = xeban t_eban_old = yeban. 439 call function 'ME_REL_EVENT_GENERAL_EBAN' in update task exporting i_call_updkz = call_updkz i_wfbn = wfbn

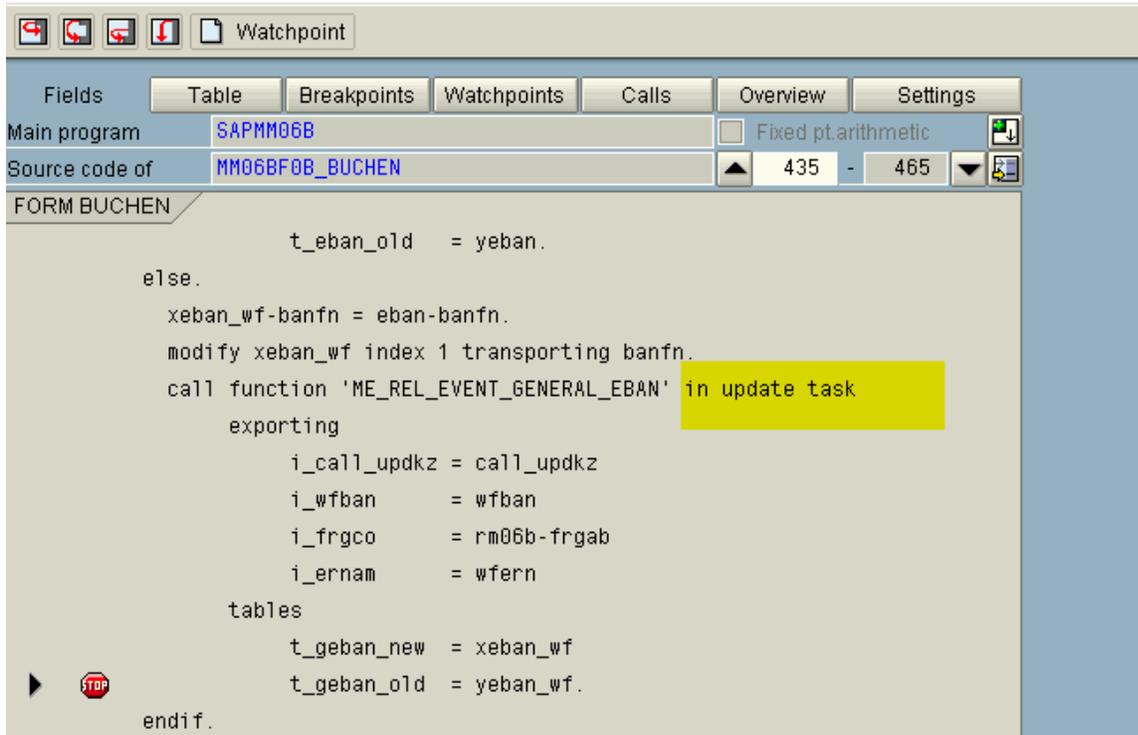
Page through the search criteria and try to select possible areas (and set break-points) where the event may be called. After a few attempts you should be able to determine function module 'ME_REL_EVENT_GENERAL_EBAN' calls the event trigger 'SWE_EVENT_CREATE'.

Note: In this example, function module 'ME_REL_EVENT_EBAN' is used for individual line item release. We are using overall line item release, i.e. we want approval based on the total value of all line items.

Notice that 'ME_REL_EVENT_GENERAL_EBAN' is called 'IN UPDATE TASK'. 'IN UPDATE TASK' is used to bundle many database changes into a single logical unit of work. All function modules belonging

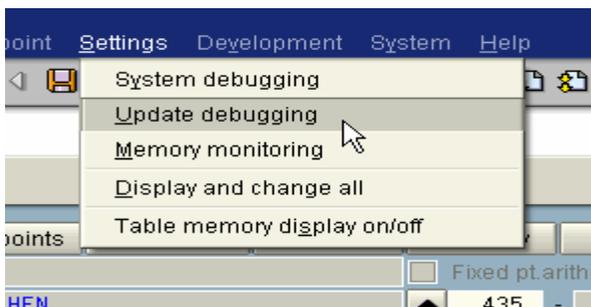
to the purchase requisition transaction are assigned an update key, when a commit work is performed; the update task reads the queue and processes all requests with the update key. We can set a breakpoint here but won't be able to debug unless we activate 'UPDATE DEBUGGING'.

ABAP_DEBUGGER



Since the ME_REL_EVENT_GENERAL_EBAN function module is being called 'in update task' we must activate update debugging.

Note: for documentation on UPDATE DEBUGGING see the R/3 online documentation: ABAP Run time Tools > Settings and Warnings (search for update debug) Settings > Update Mode...Use update debugging in debug mode...



After the purchase requisition passes all pre-processing the update task function modules are executed. Since we enabled 'Update Debugging' a new session will open with the 'Update Task' function modules. In the next print screen we finally get to the 'SWE_EVENT_CREATE' function module which is called from 'ME_REL_EVENT_GENERAL_EBAN'.

ABAP_DEBUGGER

Debugging Edit Goto Breakpoint Settings Development System Help

Main program: SAPLEBNF
 Source code of: LEBNFU14
 Fixed pt.arithmetic
 224 - 251

```

FUNCTION ME_REL_EVENT_GENERAL_EBAN
  *- ReleaseStepCreated -----*
  OBJTYPE = 'BUS2105'.
  EVENT = 'RELEASESTEP_CREATED'.
  SWC_CONTAINER BANF_CONTAINER.
  LOOP AT WFC WHERE DELKZ EQ SPACE OR
        DELKZ EQ 'Z'.
    SWC_CLEAR_CONTAINER BANF_CONTAINER.
    SWC_CREATE_CONTAINER BANF_CONTAINER.
    SWC_SET_ELEMENT BANF_CONTAINER 'ReleaseCode' WFC-FRGCO.
    OBJKEY(10) = WFC-BANFN.
  * objkey+10(5) = wfc-bnfpo.
    APPLICANT-OTYPE = 'US'.
    APPLICANT-OBJID = WFC-ERNAM.

    CALL FUNCTION 'SWE_EVENT_CREATE'
      EXPORTING
        OBJTYPE           = OBJTYPE
        OBJKEY             = OBJKEY
        EVENT              = EVENT
        CREATOR            = APPLICANT
        START_WITH_DELAY  = ''
      * IMPORTING
      *   EVENT_ID          = EVENTID
      TABLES
        EVENT_CONTAINER   = BANF_CONTAINER
      EXCEPTIONS
        OBJTYPE_NOT_FOUND = 01.
  ENDLOOP.
    
```

Internal table: WFC Type: STANDARD Format: E

	BANFN	BNFPO	FRGCO	ERNAM	DELKZ
1	0010008513	00000	EX	TNITTMANN	
1	0010008513	00000	EX	TNITTMANN	

Internal table		BANF_CONTAINER	Type	STANDARD	Format	E
1	ELEMENT	TAB_INDEX	ELEMLENGTH	TYPE	VALUE	
	RELEASECODE	000000	002	C	EX	
1	RELEASECODE	000000	002	C	EX	

At this point we see that that container element 'RELEASECODE' with a value of 'EX' is being passed.

Resolution B to Problem 2:

Review the event binding in the work item. The debug option required significant work to determine exactly where the event was being triggered for the purchase requisition release strategy. Sometimes you don't need to find the exact portion of the code that calls the event and you simply want to review the event container parameters. To do so, use the work item display transaction SWI1.

Note: since we are in a relatively technical mode as we analyze workflow issues it is recommended to switch your default workflow settings from a graphical view to a technical view. Go to the business workplace (transaction SBWP) > settings > workflow settings > personal settings > select 'Technical View' for 'Work item display' and 'Workflow log'

Personal workflow settings for Thomas Nittmann

Work item display

- User view with ActiveX (32-bit)
- User view without ActiveX
- Technical view

Workflow log

- User view with ActiveX (32-bit)
- User view without ActiveX
- Technical view

Further settings

- Display work item texts in logon language
- Enable forwarding of work items to several users
- Double-clicking on an object displays the object in the same window
- No tips & tricks in workplace
- No HTML in execution of decision tasks

Default values

Transaction SWI1 Selection Screen (Simply Execute). Consider entering date/time to reduce selection to specific workflow created for the application (purchase requisition) document. Also select 'additional data' to display the workflow number, agent, and other additional data.

Selection Report for Work Items

The screenshot shows the 'Selection Report for Work Items' interface. It has two main sections for selection criteria:

- Selection using work item ID:** A field for 'ID' with a search icon and a 'to' field with a right arrow icon.
- Selection using work item properties:** A list of properties with corresponding 'to' fields and right arrow icons:
 - Type
 - Status
 - Priority
 - Task
 - Task group
 - Date created: 27.02.2003
 - Time created: 09:25:55 to 23:59:59

At the bottom, under 'Output options', the 'Additional data' checkbox is checked.

Transaction SWI1 Output.

Work item selection

Workflow Log Icon

ID	Creation date	Creation time	Ty...	Status	Task	WF def.no.	Vers.	Work item text
547701	27.02.2003	10:25:53	F	ERROR	WS99500034	WS99500034		SAPTips: Workflow Example (Purchase Req. Release)

Display the workflow log for this item by clicking on the 'Scroll' icon (fourth icon from the left). Be sure to place the cursor on the item you want to display.

Workflow Log (View With Technical Details)

Agent: Object: Graphic: Optimize width: Choose: Save: ActiveX

Workflow: zWorkflow for overall release of req.
 Workflow instance: SAPTips: Workflow Example (Purchase Req. Release)
 Instance number: 00000547701
 Start date: 27.02.2003 Started by: WF-BATCH
 Start time: 10:25:53 Current status: Error

View: Workflow chronicle

Error	St	ID	Node number	Task	Result	Date	Time	Processing time	Object name	Object 2	Object name 2
		547701	1	SAPTips: Workflow Example (Purchase Req. Release)	Workflow started	27.02.2003	10:25:53	20m 17s			
		Thomas Nittmann		Work item created after event		27.02.2003	10:25:53		0010000525		RELEASESTEPCREATED
		Thomas Nittmann		SWP_FL_START	Purchase requisition	27.02.2003	10:25:53		USTNITTMANN		WS99500034
		Thomas Nittmann		(Sub)workflow created		27.02.2003	10:25:53		USTNITTMANN		WS99500034
		WF Manager		SWP_CALLBACK_WI_DONE		27.02.2003	10:25:54				
		WF Manager		SWP_CALLBACK_WI_DONE		27.02.2003	10:25:54				
		WF-BATCH		SWW_WI_STATUS_CHANGE	ERROR	27.02.2003	10:25:54				
				Message WL 840		27.02.2003	10:25:54				
				4 Message WL 410		27.02.2003	10:25:54				

We see that our workflow still errored. The log shows you that the workflow (WS template) started but no single step tasks (TS) have been evoked, i.e. we have an issue before we even get to the first single step task. We will resolve the error in the next couple of sessions but first let's review the event parameters. Click on the WS container icon (displayed above).

Element name	Value	Ins.	Def.
Purchase requisition	D00 800 BUS2105 0010008525	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Release code	EX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Initiator of workflow	USTNITTMANN	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Priority	5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Workflow	D00 800 FLOWITEM 000000547701	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

The container displays the event parameters passed the purchase requisition application to the workflow.

*Note: if you expect additional event parameters to be passed check the workflow definition via transaction PFTC (enter the workflow template > select 'Triggering events' tab > select binding definition button) to make sure the binding has been set-up correctly. **MOST OFTEN CUSTOM WORKFLOW EVENT PARAMETER BINDING ISSUES ARE A RESULT OF INCORRECT BINDING FROM THE EVENT TO THE WORKFLOW TASK.***

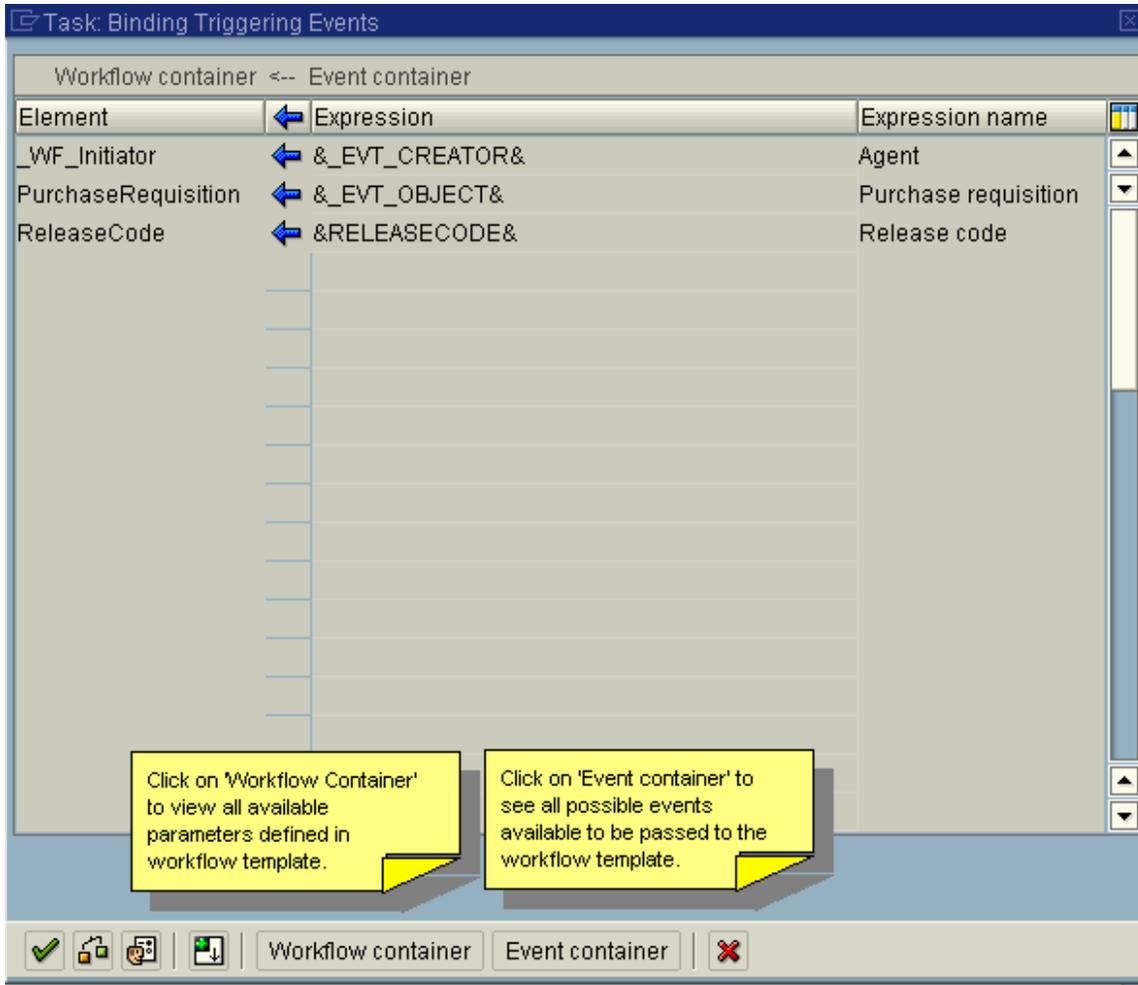
Workflow Template: Display

Workflow template: 99500034 zwf_req_rel
 Name: zWorkflow for overall release of req.
 Development class: ZFLO Appl. component: BC-BMT-WFM

Basic data Description **Triggers events** SAPphone

Standard events				
	Bindin...	Object type	Event	Name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	BUS2105	RELEASESTEP_CREATED	Purchase requisition Release step crea...
<input type="checkbox"/>	<input type="checkbox"/>			
<input type="checkbox"/>	<input type="checkbox"/>			

Binding Definitions



By clicking on the event container we see all element parameters the standard SAP purchase requisition application intended to pass to the workflow. After debugging the code where the event is triggered and reviewing the event container we see that it would be nearly impossible to add other event parameters to the workflow without modifying SAP source code OR perhaps looking for a user exit.

Note: At this point, we have analyzed the triggering mechanism of the purchase requisition release workflow. Since the trigger mechanism, in this example, is provided by SAP in the primary purchase requisition application there is no ideal way to add additional event parameters. **HOWEVER, THIS DOES NOT MEAN WE DO NOT HAVE OTHER OPTIONS. WE CAN CREATE ADDITIONAL {CUSTOM} OBJECT ATTRIBUTES OR METHODS TO PULL DATA INTO THE WORKFLOW.**

Release step created: Display Container

Element	Name	Exp.	Imp.	Man.
_EVT_OBJECT	Purchase requisition	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
_EVT_OBJTYPE	Object type	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_NAME	Event of an Object	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_OBJKEY		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_CREATOR	Agent	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_RECEIVER_ID		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_CREATION_DATE	Date and time, curre	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
_EVT_CREATION_TIME	Dat and time, curren	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ReleaseCode	Release code	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Additional Event Information:

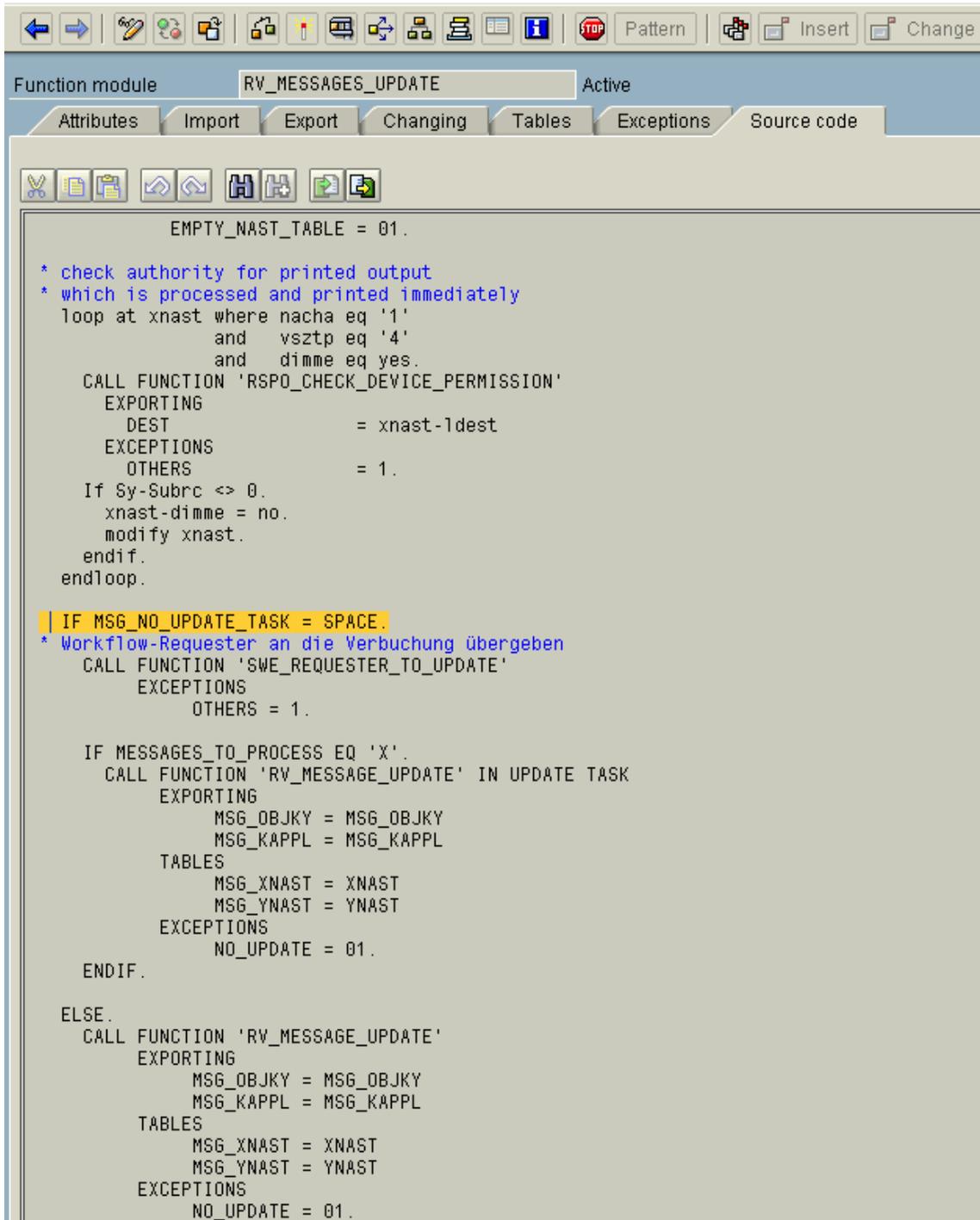
The initial part to any workflow is some sort of triggering mechanism (EVENT) to start the entire process in motion. Several standard and custom event triggering options are available to start workflow:

- The application (original SAP code) triggers workflow based on configuration settings. This is how the purchase requisition release workflow that we have been reviewing is enabled.
- Status Management, primarily used in the PP, PM, and SM modules, triggers the workflow based on either a SYSTEM status or USER status change. Why isn't status management available in other modules? Because the original developers of the other modules didn't incorporate status management into the applications.
- A custom ABAP report (and/or function module) can be written that calls when of the standard workflow event creation function modules.
- A user exit (and/or field exit) may be activated and coded to trigger workflow.
- A stand alone 'Start Transaction' can be generated via a standard tool provided by SAP. To start workflow the end user executes the transaction, fills in the selection screen, and executes to trigger the workflow.
- Business Transaction Events (BTE's) are available for FI/CO, SD, MM and other modules. BTE's must be activated via configuration (linking the BTE w/ the event function module) and/or development and are standard hooks into the application that SAP provides. When the event is triggered from the application a function module calls a BOR (Business Object Repository) event. SAP provides a function module for all BTE's that can be linked to an event.
- Change documents. If a change document object is created for the application you can link any create/change/delete change document objects to a workflow event.
- Message control (output determination) may be used in a similar fashion that print outs, faxes, etc are created. Message control is used in the MM and SD modules. Since message control is run in the background it is often difficult to debug the event creation process. We sill show an example of message control in the PURCHASE ORDER APPLICATION because purchase

requisitions do NOT use message control. Here's a solution to assist in debugging message control (this works for fax, print, workflow and all output types available in message control):

Note: Message Control, Output Determination, Output Control are different names that are generically called 'Condition Technique', i.e. don't let SAP confuse you - they all refer to the same process.

- a. Message control (aka output determination) uses function module 'RV_MESSAGES_UPDATE' to generate output IF the configuration is set-up properly.
- b. SET A BREAK POINT AT 'IF MSG_NO_UPDATE_TASK = SPACE.' statement in the function module 'RV_MESSAGE_UPDATE' (line 95 in R/3 46C).

Function Builder: Display RV_MESSAGES_UPDATE


```

EMPTY_NAST_TABLE = 01.

* check authority for printed output
* which is processed and printed immediately
loop at xnast where nacha eq '1'
    and vsztp eq '4'
    and dimme eq yes.
    CALL FUNCTION 'RSPO_CHECK_DEVICE_PERMISSION'
        EXPORTING
            DEST           = xnast-ldest
        EXCEPTIONS
            OTHERS        = 1.
    If Sy-Subrc <> 0.
        xnast-dimme = no.
        modify xnast.
    endif.
endloop.

| IF MSG_NO_UPDATE_TASK = SPACE.
* Workflow-Requester an die Verbuchung übergeben
CALL FUNCTION 'SWE_REQUESTER_TO_UPDATE'
    EXCEPTIONS
        OTHERS = 1.

IF MESSAGES_TO_PROCESS EQ 'X'.
    CALL FUNCTION 'RV_MESSAGE_UPDATE' IN UPDATE TASK
        EXPORTING
            MSG_OBJKY = MSG_OBJKY
            MSG_KAPPL = MSG_KAPPL
        TABLES
            MSG_XNAST = XNAST
            MSG_YNAST = YNAST
        EXCEPTIONS
            NO_UPDATE = 01.
ENDIF.

ELSE.
CALL FUNCTION 'RV_MESSAGE_UPDATE'
    EXPORTING
        MSG_OBJKY = MSG_OBJKY
        MSG_KAPPL = MSG_KAPPL
    TABLES
        MSG_XNAST = XNAST
        MSG_YNAST = YNAST
    EXCEPTIONS
        NO_UPDATE = 01.

```

Note: Several steps are required to set-up message control for workflow. In this example Output Medium '9' (Events SAP Business Workflow) was set-up for output type 'NEU' (Purchase Orders) via transaction 'M/34'. Output type '9' is used to create an event; the other option is to set-up output medium 'T' and trigger a task directly. Program 'RVNSWE01' and form routine 'CREATE_EVENT' is used to generate the event. Custom sub-type ZBUS2012 was created,

delegated to ZBUS2012, and custom event 'zMessageWFEvent' was created. Template workflow (WS99500035) was created with a simple task (TS99500021) with a method that has 'break-point' in it. Lastly, a condition record (MN04) was created for document type 'NB' to automatically generate the workflow output when the purchase order is saved.

- c. Determine the form routine used by message control to create the workflow event. Message Control (output determination) for purchase orders is set-up in the IMG under MM > Purchasing > Messages > Output Control. We simply need to review the message type 'NEU' for purchase order output to find the form driving program and form routine.

Display View "Processing routines": Overview

Medium	Short text	Program	FORM routine	Form
1	Print output	SAPFM06P	ENTRY_NEU	MEDRUCK
2	Fax	SAPFM06P	ENTRY_NEU	MEDRUCK
6	EDI	RSNASTED	EDI_PROCESSING	
9	Events (SAP Business Workflow)	RVNSWE01	CREATE_EVENT	
A	Distribution (ALE)	RSNASTED	ALE_PROCESSING	

Hint: SAP uses the condition technique for pricing, output, account determination, and many other functions. Each application has a different transaction to configure the condition technique. However, you can use transaction VK01 to select any condition technique you would like to maintain.

- d. Set a break-point in the form routine 'CREATE_EVENT'.

ABAP Editor: Display Report RVNSWE01

```

REPORT RVNSWE01.
TABLES: NAST.

*-----*
*      FORM CREATE_EVENT      *
*-----*
FORM CREATE_EVENT USING RETURNCODE
                    US_SCREEN.
* das anwendungsunspezifische Verarbeitungsprogramm ist RSWEMC01
PERFORM CREATE_EVENT(RSWEMC01) USING RETURNCODE
                    US_SCREEN.
ENDFORM.

```

- e. Go to the application that uses message control and create an application document, e.g. transaction 'ME21' for purchase orders (the purchase requisition application does NOT use message control). You need to make certain a condition record (MN04) is set-up to automatically propose the workflow output.

Create Purchase Order : Item Overview

Item	Material	Short text	PO quantity	O...	C Deliv. date	Net price	Per	O...	Matl group	Plnt	SLoc
10	R100028	SAPTips Workflow Message Control		6 PC	04.03.2003	75,001		PC	R1114	R310	0001
20					D						

- f. When you save the purchase order a debug session starts for function module 'RV_MESSAGES_UPDATE'.

Note: The debug session will start only if message control automatically proposed output. If a debug session did not open then manually add the out put type (Headers > Messages) and save.

ABAP_DEBUGGER

Fields Table Breakpoints Watchpoints Calls Overview Settings

Main program SAPLV61B Fixed pt.arithmetic

Source code of LV61BU04 83 - 113

```

FUNCTION RV_MESSAGES_UPDATE
    and dimme eq yes.
    CALL FUNCTION 'RSPO_CHECK_DEVICE_PERMISSION'
    EXPORTING
        DEST           = xnast-ldest
    EXCEPTIONS
        OTHERS        = 1.
    If Sy-Subrc <> 0.
        xnast-dimme = no.
        modify xnast.
    endif.
endloop.

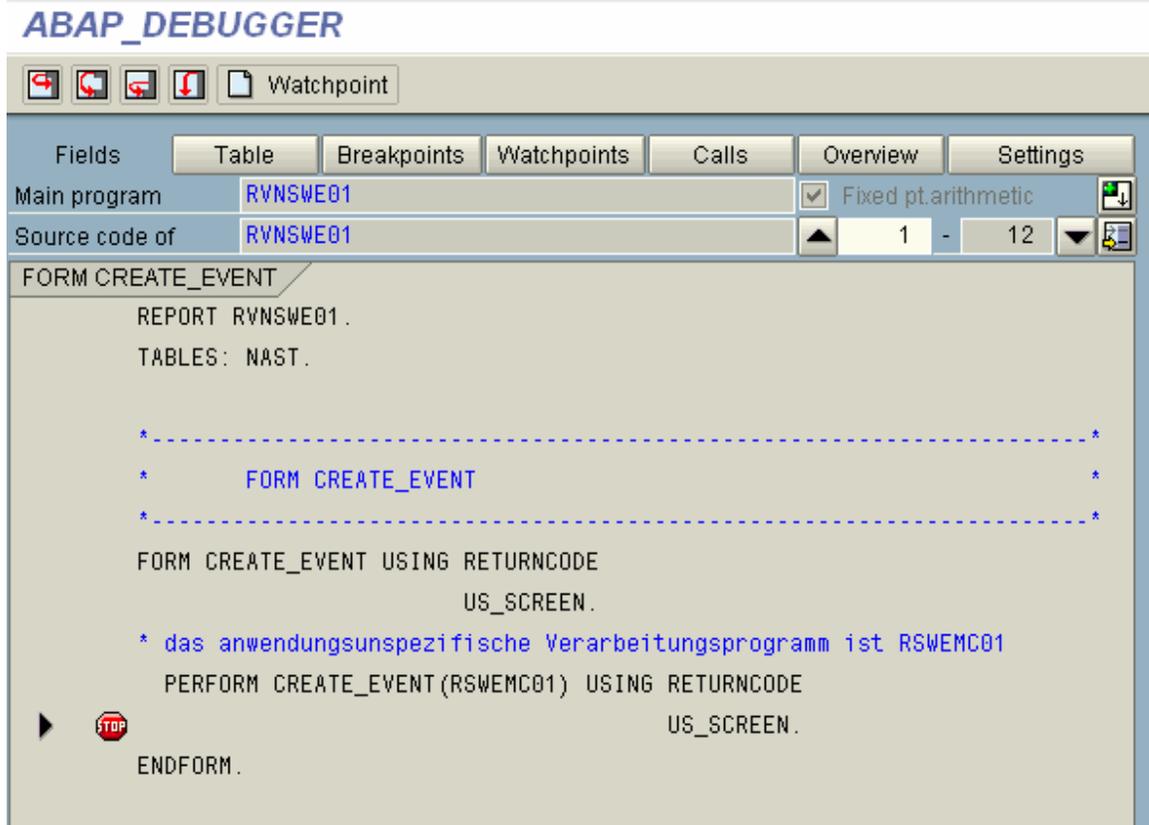
▶  IF MSG_NO_UPDATE_TASK = SPACE.
    * Workflow-Requester an die Verbuchung übergeben
    CALL FUNCTION 'SWE_REQUESTER_TO_UPDATE'
    EXCEPTIONS
        OTHERS = 1.

    IF MESSAGES_TO_PROCESS EQ 'X'.
        CALL FUNCTION 'RV MESSAGE UPDATE' IN UPDATE TASK
    
```

- g. Set variable 'MSG_NO_UPDATE_TASK' = 'X' and continue (be sure to change it by clicking on the pencil).



- h. When you continue, SAP will break at the form routine for the creation of the event.



- i. Later we will review how to continue debugging at the workflow task level.

Agent Resolution Debugging

All workflow tasks that generate work items (foreground tasks) need to be assigned to an individual or group of agents.

Problem 3 Agent Resolution:

A workflow work item is created but it is not assigned to an agent.

Resolution to Problem 3:

First, review the work item workflow log via transaction SWI1 (click on workflow log 'Scroll' Icon).

Work item selection

ID	Creation date	Creation time	Type	Status	Task	WF def.no.	Work item text
547719	27.02.2003	17:44:39	F	ERROR	WS99500034	WS99500034	SAPTips: Workflow Example (Purchase Req. Release)

Workflow Log (View With Technical Details)

Workflow: zWorkflow for overall release of req.
 Workflow instance: SAPTips: Workflow Example (Purchase Req. Release)
 Instance number: 000000547719
 Start date: 27.02.2003 Started by: WF-BATCH
 Start time: 17:44:39 Current status: **Error**

View: Workflow chronicle

Error	St	ID	Node number	Task	Result	Date	Time	Processing time	Object name
Error Agent				Executed action		Date	Time		Object 2
		547719	1	SAPTips: Workflow Example (Purchase Req. Rel)	Workflow started	27.02.2003	17:44:39	5m 32s	
	Thomas Nittmann			SWW_FI_START	Work item created after event	27.02.2003	17:44:39		Purchase requisition 0010008526
	Thomas Nittmann			(SUB)workflow created		27.02.2003	17:44:39		USTNITTMANN
	WF Manager			SWP_CALLBACK_WI_DONE		27.02.2003	17:44:40		WS99500034
	WF Manager			SWP_CALLBACK_WI_DONE		27.02.2003	17:44:40		
	WF-BATCH			SWW_WI_STATUS_CHANGE		27.02.2003	17:44:40		ERROR
				Message WL 840		27.02.2003	17:44:40		
			4	Message WL 410		27.02.2003	17:44:40		

Review the log and pinpoint where the error occurs. Double click where you see the first 'Message ...' occurs'.

Workflow Log (View With Technical Details)

Workflow: zWorkflow for overall release of req.
 Workflow instance: SAPTips: Workflow Example (Purchase Req. Release)
 Instance number: 000000547719
 Start date: 27.02.2003 Started by: WF-BATCH
 Start time: 17:44:39 Current status: **Error**

View: Workflow chronicle

Error	St	ID	Node number	Task	Result	Date	Time	Object	Object name
Error Agent				Executed action		Date	Time	Object	Object name
		547719	1	SAPTips: Workflow Example (Purchase Req. Rel)	Workflow started	27.02.2003	17:44:39	Purchase requisition 0010008526	
	Thomas Nittmann			SWW_FI_START	Work item created after event	27.02.2003	17:44:39	USTNITTMANN	USTNITTMANN
	Thomas Nittmann			(SUB)workflow created		27.02.2003	17:44:39	USTNITTMANN	USTNITTMANN
	WF Manager			SWP_CALLBACK_WI_DONE		27.02.2003	17:44:40		
	WF Manager			SWP_CALLBACK_WI_DONE		27.02.2003	17:44:40		
	WF-BATCH			SWW_WI_STATUS_CHANGE		27.02.2003	17:44:40		ERROR
				Message WL 840					
			4	Message WL 410					

Double click here to see error message. Usually click where the first message exists.

Click the long text ('?' icon)

Information

Work item 547720: Agents of type A cannot be defined

✓ ?

The long text of the issues describes that an agent could not be determined.

Performance Assistant

Work item 547720: Agents of type A cannot be defined

Message no. WL 840

Diagnosis

The agents of the specified type could not be determined for the specified work item.

The following agent types exist:

- A Current agent of a work queue work item
- D Work item recipient for missed latest end
- E Excluded agent
- L Work item recipient if the latest start is missed
- N Message recipient for completion
- W Work item recipient if requested end is missed

System Response

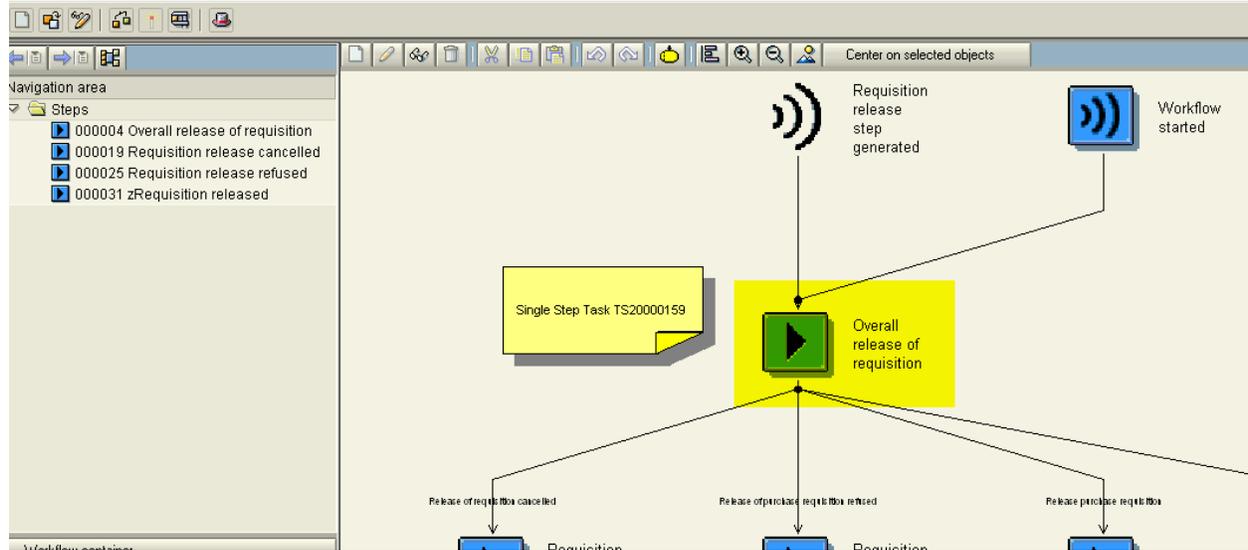
The action was terminated.

Procedure

Check the organizational assignment of the underlying task. If applicable, also check the role entered there and the function module implementing it.
The excluded agents can only be determined via the workflow definition. Therefore check this definition.

We need to review the standard SAP workflow agent resolutions before we continue. The first single step task (TS) step of the purchase requisition workflow (WS) release strategy is '000004' (TS20000159).

Workflow Builder - Display 'zwf_req_rel' [Active,Saved]



You can define agent resolution at the step (000004) level. Any agent resolution defined at this level will be superseded if agent resolution at the single step (TS) task level.

Step Edit Goto Extras System Help

Workflow Builder - Display 'zwf_req_rel' [Active,Saved]

Double click on task TS20000159 and select the 'default role' tab. A standard role (AC20000026) is defined. THIS IS USED TO DETERMINE THE AGENT FOR THE OVERALL RELEASE. HERE'S WHERE OUR PROBLEM OCCURS. WE NEED TO FIGURE OUT WHY AN ERROR IS OCCURRING HERE.

Standard Task: Display

Standard task: 20000159 mm_req_rel_c
 Name: Overall release of requisition
 Development class: ME Appl. component: MM-PUR

Alternative methods | Triggers events | Terminating events | **Default roles** | SAPph...

Type of role	Binding	Standard r...	Role name
Agent (Default Role)	<input checked="" type="checkbox"/>	20000026	Person responsible for requis. release
Recipient for Missed Latest Start	<input type="checkbox"/>	00000000	
Recipient for Missed Latest End	<input type="checkbox"/>	00000000	

You can double click on the standard role (AC2000026) to launch the PFAC transaction and review the function module that determines the agent.

Standard Role: Display

Standard role: 20000026 MM_req_rel_c
 Name: Person responsible for requis. release
 Development class: ME Appl. component: MM-PUR

Role definition | Description | Container

Basic data

Abbr.: MM_req_rel_c
 Name: Person responsible for requis. release

Role definition

Type: Function to be executed
 Function module: **ME_REL_GET_RESPONSIBLE**
 Terminate if role resolution has no result

Set a SOFT break-point in ME_REL_GET_RESPONSIBLE (use transaction SE37).

```

FUNCTION me_rel_get_responsible.
*-----
**Lokale Schnittstelle:
**
**   TABLES
**       ACTOR_TAB STRUCTURE SWHACTOR
**       AC_CONTAINER STRUCTURE SWCONT
**
**   EXCEPTIONS
**       NOBODY_FOUND
**
**-----
** 020198 H91758 fehlender Tabellenparameter ac_container im User-Exit
**
*-----
INCLUDE <contain>.
TABLES: eban, ekko, t16fw.
*data: eban-object type swc_object.
DATA: object TYPE swc_object.
DATA: h_object_ext LIKE rhobjects-object.          "108391
DATA: BEGIN OF ebankey,
       banfn LIKE eban-banfn,
       bnfpo LIKE eban-bnfpo,
       END OF ebankey.

DATA: BEGIN OF ekkokey,
       ebeln LIKE ekko-ebeln,
       END OF ekkokey.
* Initialisieren der Tabelle ACTOR_TAB
REFRESH actor_tab.
CLEAR actor_tab.
swc_get_element ac_container 'ReleaseCode' rm06b-frgab.
swc_get_element ac_container 'requisition' object.
    
```

Now, we'll use the workflow test transaction SWUS to execute requisition release workflow WS99500034. We will need the container elements so we should go to SWI1 (and then workflow log > container elements) first and displayed the errored workflow event parameters.

Workflow Log (View With Technical Details)

Workflow: zWorkflow for overall release of req.
 Workflow instance: SAPTips: Workflow Example (Purchase Req. Release)
 Instance number: 000000547719
 Start date: 27.02.2003 Started by: WF-BATCH
 Start time: 17:44:39 Current status: **Error**

View: Workflow chronicle

Error	St	ID	Node number	Task	Result
Error Agent			Executed action	Date	Time
		547719	1	SAPTips: Workflow Example (Purchase Req. Rel	Workflow
				Message WL 840	
			4	Message WL 410	

Element name	Value	Ins.	Def.
Purchase requisition	D00 800 BUS2105 0010008526	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Release code	EX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Initiator of workflow	USTNITTMANN	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Priority	5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Workflow	D00 800 FLOWITEM 000000547719	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Go to SWUS and execute the workflow with the above container elements.

Start Workflow (Test Environment)

Task:

Task type: Workflow template

Name: zWorkflow for overall release of req.

Validity: 01.01.1900 To 31.12.9999

Runtime data

Input data set

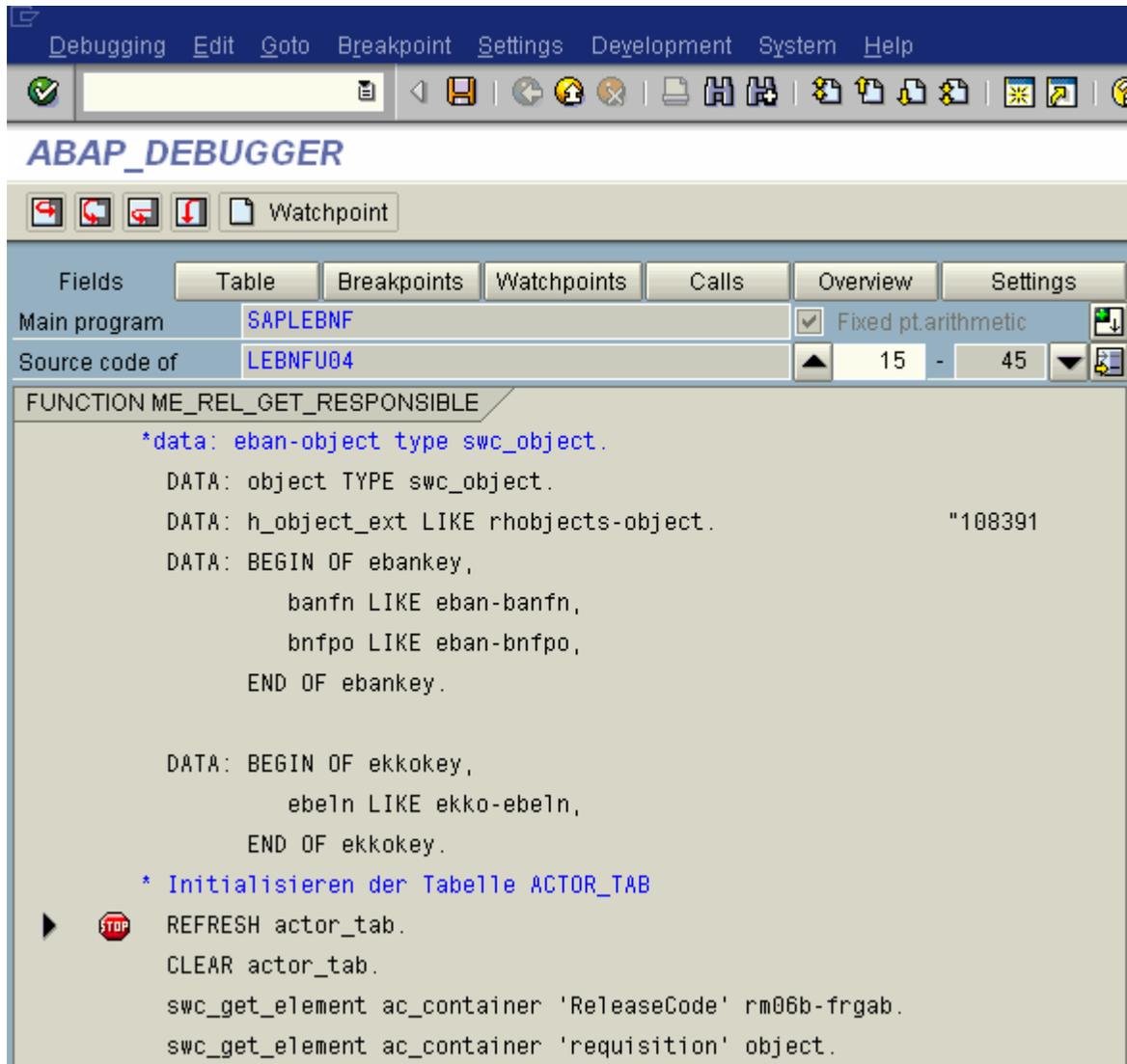
Deadline data set

Select 'Input Data' and enter the container elements

Element name	Value	Ins.	Def.
Purchase requisition	BUS2105 0010008526	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Release code	EX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Be sure to use the pull down for the 'Purchase requisition' element because it is defined as an object type

Execute the test workflow. The debug point set-up in function module ME_REL_GET_RESPONSIBLE will be evoked and you can start debugging the issue.



Run through the debugger and you will eventually get to a function module where SYST-SUBRC = 1.

ABAP_DEBUGGER

Fields | Table | Breakpoints | Watchpoints | Calls | Overview | Settings

Main program: SAPLRHWA | Fixed pt.arithmetic

Source code of: LRHWAU01 | 188 - 218

FUNCTION RH_GET_ACTORS

```

call function 'RH_TASK_AGENT_CHECK'
  exporting
    org_otype      = actor_tab-otype
    org_objid      = actor_tab-objid
    task_otype     = task-otype
    task_objid     = task-objid
    act_wi_id      = act_wi_id
    act_plvar      = act_plvar
    act_begda      = search_date
    act_endda      = search_date

  tables
    excluded_agents = excluded_agents

  exceptions
    others          = 1.

if sy-subrc > 0.
  error_tab = actor_tab.
  collect error_tab.

  delete actor_tab.
endif.
endloop.

describe table actor_tab lines actor_lines.
if actor_lines = 0.
  if exec_enforce is initial.
    message s313(5w) with wf_object act_task
      raising no_valid_agent_determined.
  else.
    message e313(5w) with wf_object act_task
      raising no_valid_agent_determined.
  endif.
endif.

```

Field names | 1 - 4 | Field contents

t16fw-otype	
t16fw-objid	
SY-SUBRC 1	SY-TABIX 1
	SY-DBCNT 0

The function module 'RH_TASK_AGENT_CHECK' determines if the task is assigned to a user id. It isn't so we receive the error. Let's briefly discuss task 'ATTRIBUTES'.

- A task w/ attribute 'General task' indicates any SAP user can execute the task.
- A task w/ attribute 'General forwarding allowed' indicates that the original agent may forward the task to any user (even if they are not assigned to the task).
- A task w/ attribute 'General forwarding not allowed' indicates that the work item can only be forwarded to agents assigned to the task.

To determine the task attributes go to transaction PFTC > enter the workflow template > go to display > select workflow builder >

Task: Maintain

Task type	Workflow template
Task	99500034 zwf_req_rel
Name	zWorkflow for overall release of req

Workflow Template: Display

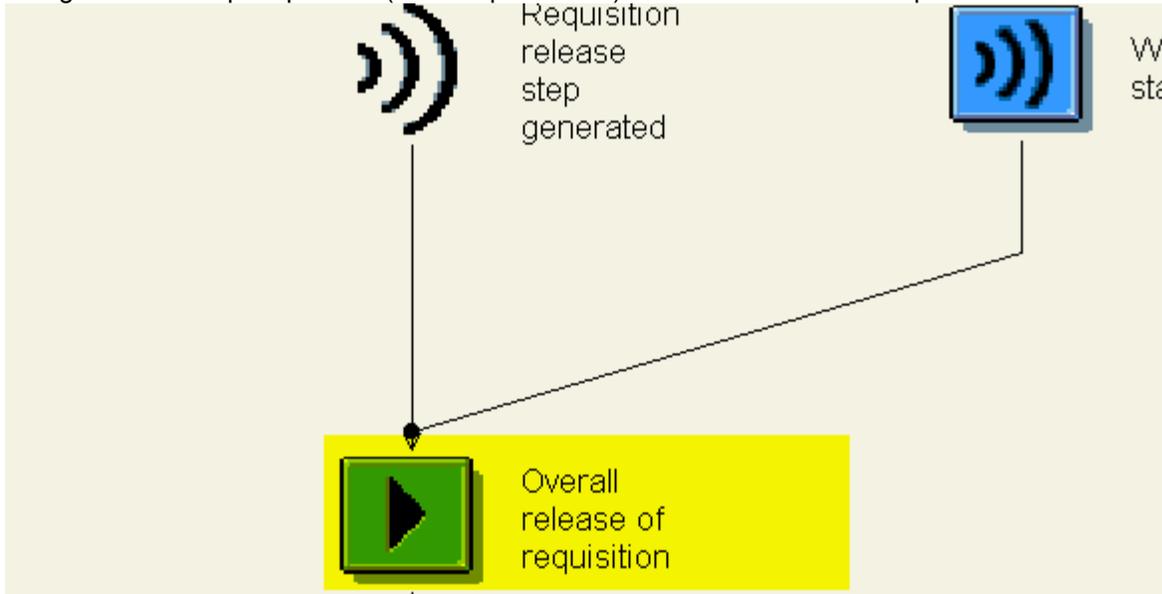
Workflow template	99500034	zwf_req_rel
Name	zWorkflow for overall release of req.	
Development class	ZFLO	Appl. component

Name	
Abbr.	zwf_req_rel
Name	zWorkflow for overall release of req.
Work item text	SAPTips: Workflow Example (Purchase Req. Release)
Release status	Not defined

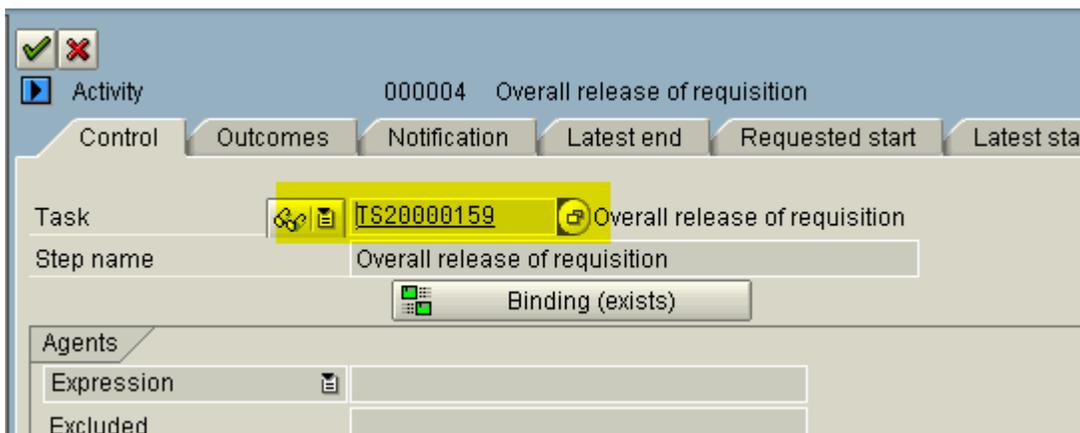
Workflow definition

Workflow definition Workflow Builder

Navigate to the step in question (here step '000004') > double click on the step >

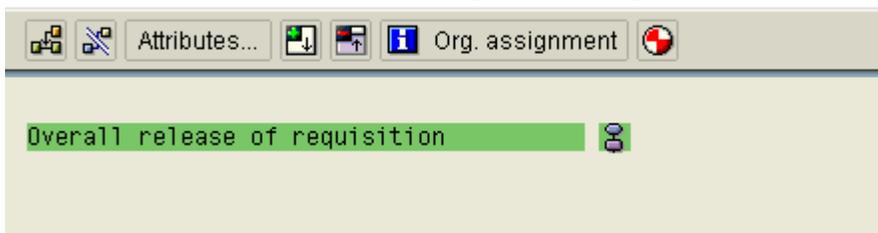


Double click on TS20000159 >

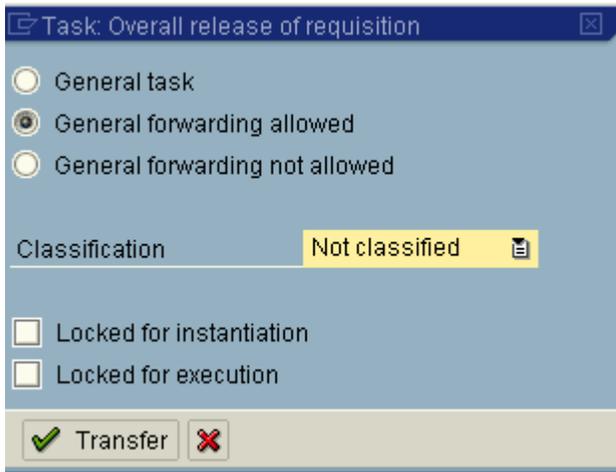


From the menu select Additional Data > Agent Assignment > Maintain >

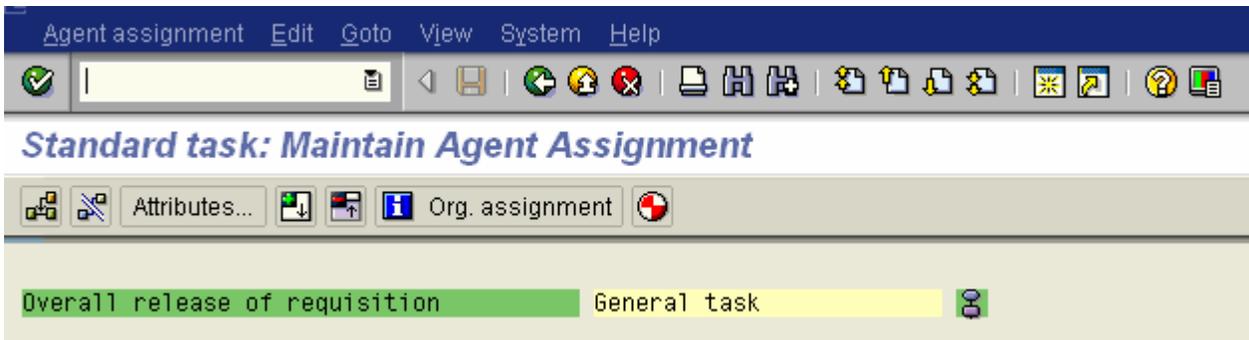
Standard task: Maintain Agent Assignment



Select the line item and then click attributes >



This is our issue. The task is not classified as general. Option one is to create an org. structure, assign a position, assign a user (or person), and then assign the task to the position. Or we can make the task general so it can be executed by anybody (with the expectation that the OMGQ configuration role resolution will be used). Let's change the assignment to 'General Task' and save.



Re-execute the SWUS test transaction. We notice we no longer have an issue w/ RH_TASK_AGENT_CHECK

The screenshot displays the SAP ABAP Debugger interface. At the top, there is a menu bar with options: Debugging, Edit, Goto, Breakpoint, Settings, Development, System, and Help. Below the menu is a toolbar with various icons for navigation and execution. The main window is titled "ABAP_DEBUGGER" and contains several tabs: Fields, Table, Breakpoints, Watchpoints, Calls, Overview, and Settings. The "Fields" tab is active, showing the current state of variables.

The main program is identified as "SAPLRHWA" and the source code of "LRHWAU01" is being viewed. The current line of code is 188, and the range is from 188 to 218. The function being debugged is "FUNCTION RH_GET_ACTORS".

```

call function 'RH_TASK_AGENT_CHECK'
  exporting
    org_otype      = actor_tab-otype
    org_objid      = actor_tab-objid
    task_otype     = task-otype
    task_objid     = task-objid
    act_wi_id      = act_wi_id
    act_plvar      = act_plvar
    act_begda      = search_date
    act_endda      = search_date
  tables
    excluded_agents = excluded_agents
  exceptions
    others          = 1.

if sy-subrc > 0.
  error_tab = actor_tab.
  collect error_tab.

  delete actor_tab.
endif.
endloop.

describe table actor_tab lines actor_lines.
if actor_lines = 0.
  if exec_enforce is initial.
    message s313(5w) with wf_object act_task
      raising no_valid_agent_determined.
  else.
    message e313(5w) with wf_object act_task
      raising no_valid_agent_determined.
  endif.
endif.

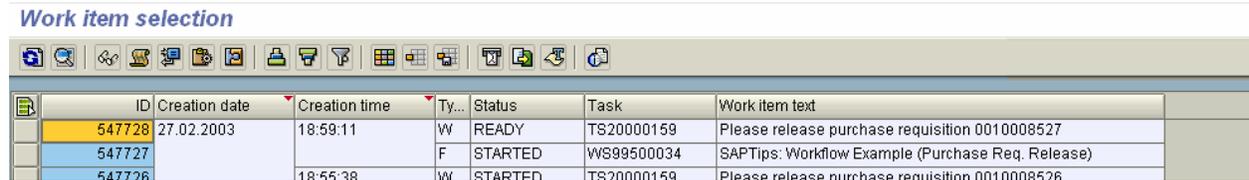
```

At the bottom of the debugger, there is a "Field names" section with a dropdown menu showing "1 - 4" and a "Field contents" section with a table of variable values:

Field names	Field contents
SY-SUBRC 0	
SY-TABIX 1	
SY-DBCNT 1	

Create another purchase requisition from scratch and validate the workflow is correctly created (SWI1).

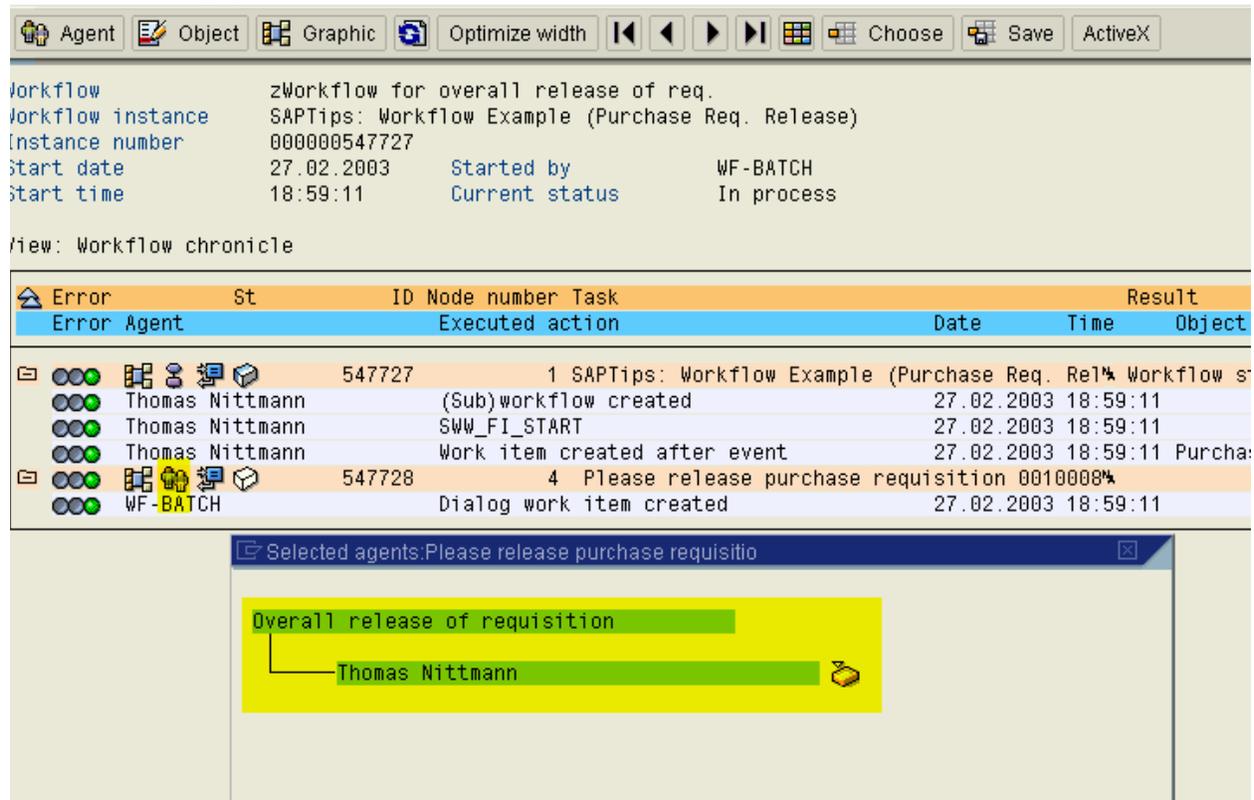
Work item selection



ID	Creation date	Creation time	Ty...	Status	Task	Work item text
547728	27.02.2003	18:59:11	W	READY	TS20000159	Please release purchase requisition 0010008527
547727			F	STARTED	WS99500034	SAPTips: Workflow Example (Purchase Req. Release)
547726		18:55:38	W	STARTED	TR20000159	Please release purchase requisition 0010008526

By clicking on the agent icon you will notice that an agent is now correctly assigned and the workflow is no longer in error.

Workflow Log (View With Technical Details)



Workflow zWorkflow for overall release of req.
 Workflow instance SAPTips: Workflow Example (Purchase Req. Release)
 Instance number 000000547727
 Start date 27.02.2003 Started by WF-BATCH
 Start time 18:59:11 Current status In process

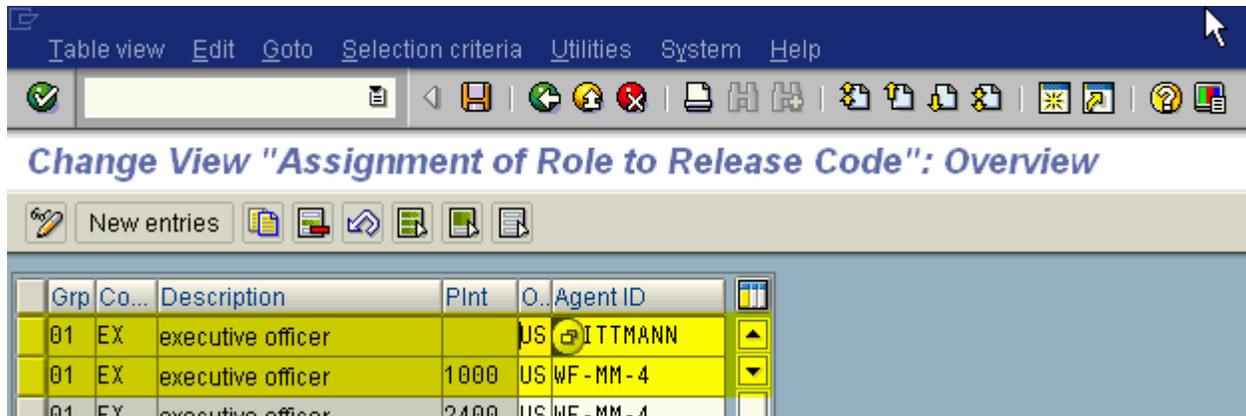
View: Workflow chronicle

Error	St	ID	Node number	Task	Result
Error Agent			Executed action	Date	Time Object
Thomas Nittmann		547727	1	SAPTips: Workflow Example (Purchase Req. Rel%	Workflow s
Thomas Nittmann			(Sub)workflow created	27.02.2003	18:59:11
Thomas Nittmann			SWW_FI_START	27.02.2003	18:59:11
Thomas Nittmann			Work item created after event	27.02.2003	18:59:11
WF-BATCH		547728	4	Please release purchase requisition 0010008527	Purcha:
WF-BATCH			Dialog work item created	27.02.2003	18:59:11

Selected agents: Please release purchase requisitio

Overall release of requisition
 Thomas Nittmann

Note: a common migration issue specific to purchase requisition releases occurs when clients transition from individual line item release to overall release. If OMGQ workflow configuration is maintained at the plant level (based on single line item release) the workflow will error for overall release because there is NO PLANT AT THE HEADER LEVEL OF THE PURCHASE REQUISITION, i.e. you must remove the plant from OMGQ > Workflow. By using the same debugging principles just described you would be able to determine the error.



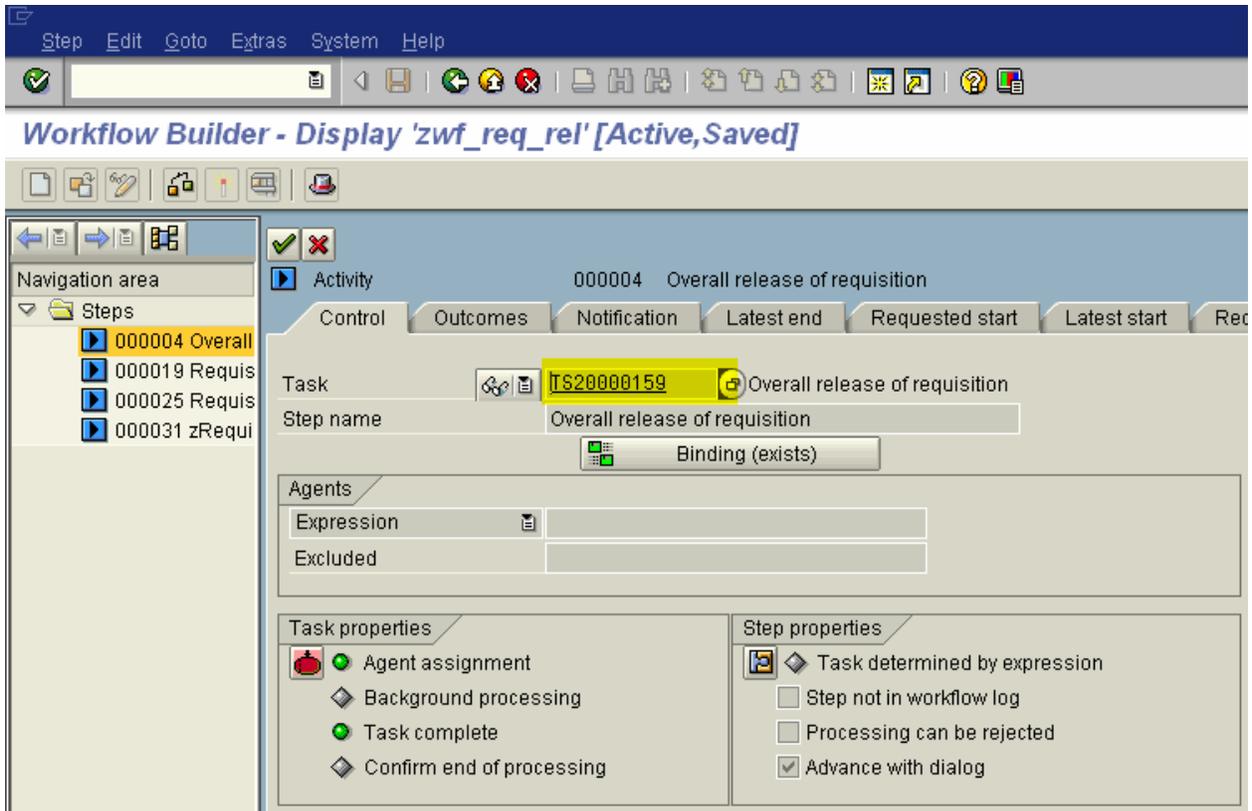
Workflow Task (Method) Debugging

The code for a single step task (TS) is written in methods assigned to objects in the Business Object Repository (BOR). It is often necessary to debug the method code to resolve workflow issues. Ideally, you want to start the debugger for the entire workflow but break at some point in the custom method code. Let's review the process for debugging BOTH foreground (dialog) and background (synchronous) single step tasks.

Foreground (Dialog) Tasks

We will start by dissecting the first foreground task in the custom purchase requisition workflow WS99500034 we created. The first task is the overall release of the requisition (TS20000159). Use the workflow builder (transaction PFTC > enter 'Workflow template' for task type > enter task number '99500034' for Name > select display > click on the 'Workflow builder' button) to call up the workflow and double click on the first task (step '000004') in the process flow diagram to launch the workflow template task detail screen.

The goal is to set a break-point in the first dialog task to aid in debugging any issues.



Next, double click on the task to launch the task details.

Standard Task: Display

Container Initial values

Standard task: 20000159 mm_req_rel_c
 Name Overall release of requisition
 Development class ME Appl. component MM-PUR

Basic data Description Alternative methods Triggering events Terminating events

Name
 Abbr. mm_req_rel_c
 Name Overall release of requisition
 Work item text Please release purchase requisition & _WI_Object_Id.Number&
 Release status Not defined

Object method
 Object type BUS2105 Purchase requisition
 Method SINGLERELEASE Individual release
 Synchronous object method
 Object method with dialog

Execution
 Background processing Executable as form
 Confirm end of processing

Double click on method 'SINGLERELEASE' to launch the BOR definition of the method.

Display Object Type ZBUS2105

Object type ZBUS2105 SAPTips Purchase Requisition

- Interfaces
- Key fields
- Attributes
- Methods
 - zPurchaseRequisition.CreateFromData ✓ Create purchase requisition
 - zPurchaseRequisition.Change ✓ Change purchase requisition
 - zPurchaseRequisition.GetItems ✓ Read purchase requisition item
 - zPurchaseRequisition.GetItemsForRelease ✓ List purchase requisitions awaiting release
 - zPurchaseRequisition.Delete ✓ Delete/close purchase requisition
 - zPurchaseRequisition.GetDetail ✓ Display purchase requisition details
 - zPurchaseRequisition.GetReleaseInfo ✓ Detailed release information on purchase requisition
 - zPurchaseRequisition.Release ✓ Release purchase requisition
 - zPurchaseRequisition.ResetRelease ✓ Cancel release of purchase requisitions
 - zPurchaseRequisition.SingleRelease ✓ Individual release**
 - zPurchaseRequisition.ExistenceCheck ✓ Check existence of object
 - zPurchaseRequisition.Display ✓ Display object
 - zPurchaseRequisition.InfoReleaseReset ✓ Info: Release cancelled
 - zPurchaseRequisition.Edit ✓ Process purchase requisition
 - zPurchaseRequisition.InfoReleaseRejected ✓ Info: Release refused
 - zPurchaseRequisition.InfoReleaseEffectuated ✓ Info: Release effected
 - zPurchaseRequisition.zNotifyReqRelease zNotification of Requisition Release

Notice that object 'ZBUS2105' was called. This is fine. Custom sub-type 'ZBUS2105' was created and delegated to super type 'BUS2105'. Place the cursor on the method 'zPurchaseRequisition.SingleRelease' and then click the 'Program' button to launch the code for this method. We will set a soft break-point.

Object Type: Editor Display Program RBUS2105

```

235 begin_method singlerelease changing container.
236 DATA:
237     purchaserequisition LIKE eban-banfn,
238     releasecode LIKE rm06b-frgab.
239 DATA: call_updkz,
240     auto.
241 DATA: BEGIN OF bat OCCURS 10.
242     INCLUDE STRUCTURE eban.
243 DATA: updkz.
244 DATA: END OF bat.
245 TABLES: rm06b.
246 swc_get_element container 'ReleaseCode' releasecode.
247 * SET PARAMETER ID 'BAN' FIELD OBJECT-KEY-PURCHASEREQUISITION.
248 CLEAR rm06b-bnfpo.
249 SET PARAMETER ID 'BAP' FIELD rm06b-bnfpo.
  
```

We cannot simply create a new purchase requisition (ME51) and expect the break-point to hit because workflow is based on tRFC (transactional RFC) engine, i.e. it's an asynchronous RFC (that ensures unique execution). To test we will use the workflow test transaction 'SWUS' and a former purchase requisition (00100008529) that was already created and has a release strategy.

Start Workflow (Test Environment)

Refresh organizational environment | Workflow log | Business Workplace

Task: WS99500034 zwf_req_rel
 Task type: Workflow template
 Name: zWorkflow for overall release of req.
 Validity: 01.01.1900 To 31.12.9999

Runtime data
 Input data set
 Deadline data set

Input data
 Deadline data

Initialize Container

Element name	Value
Purchase requisition	BUS2105 0010008529
Release code	EX

When we execute the data (after adding the input data) the method code is available for debugging.

ABAP_DEBUGGER

Watchpoint

Fields Table Breakpoints Watchpoints Calls Overview Settings

Main program RBUS2105 Fixed pt.arithmetic

Source code of RBUS2105 236 - 266

FORM SINGLERELEASE

```

DATA:
  purchaserequisition LIKE eban-banfn,
  releasecode LIKE rm06b-frgab.

DATA: call_updkz,
      auto.

DATA: BEGIN OF bat OCCURS 10.
      INCLUDE STRUCTURE eban.

DATA: updkz.

DATA: END OF bat.

TABLES: rm06b.

swc_get_element container 'ReleaseCode' releasecode.
* SET PARAMETER ID 'BAN' FIELD OBJECT-KEY-PURCHASEREQUISITION.
▶  CLEAR rm06b-bnfpo.
  SET PARAMETER ID 'BAP' FIELD rm06b-bnfpo.
  * SET PARAMETER ID 'BFC' FIELD RELEASECODE.
  * CALL TRANSACTION 'ME54' AND SKIP FIRST SCREEN.
  *-- Aufruf Einzelfreigabe -----*
REFRESH bat.
CLEAR bat.
call_updkz = 'W'.
auto = 'X'.
rm06b-refbn = object-key-number.

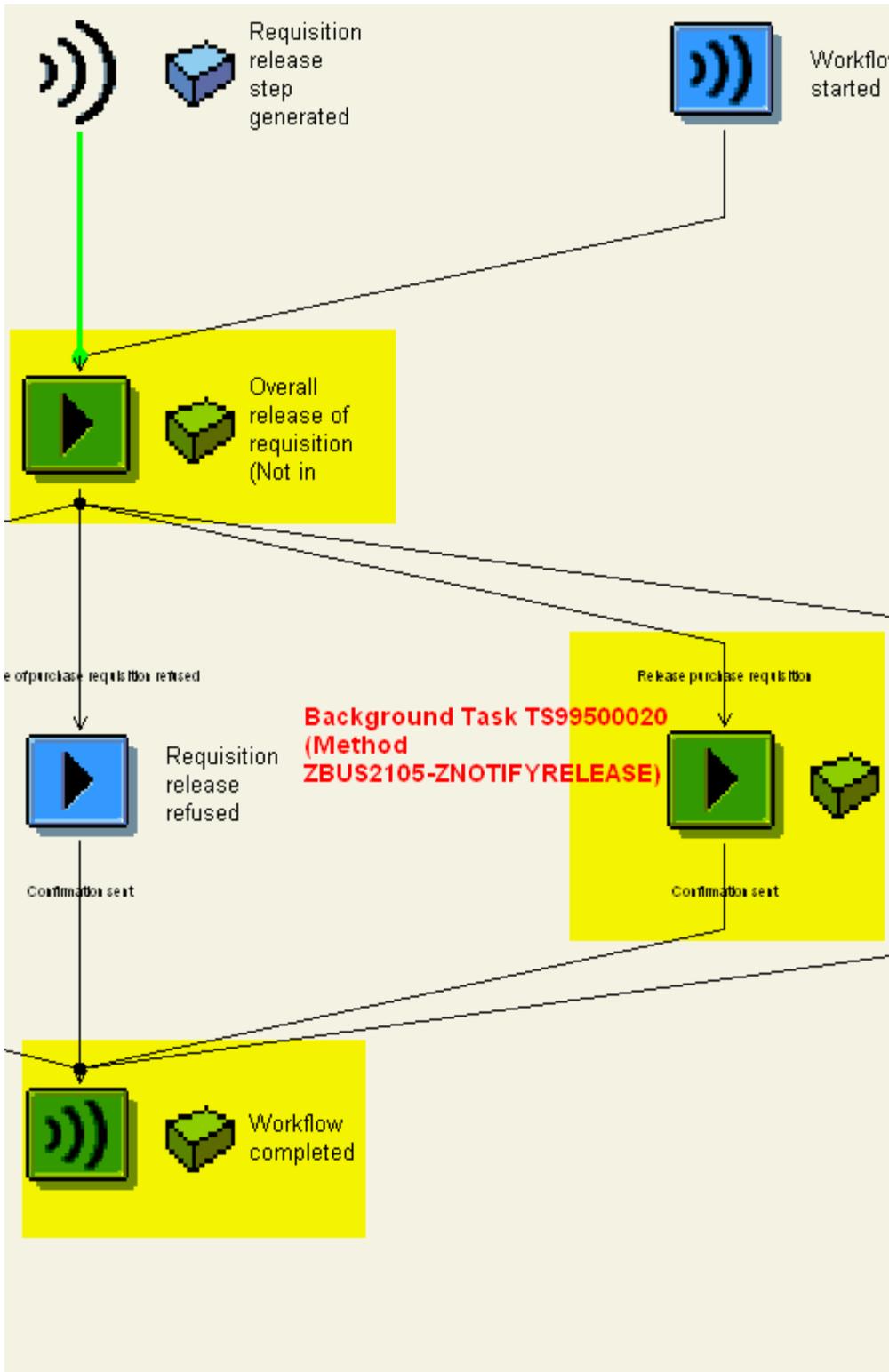
```

Background (Synchronous) Tasks

Special consideration is required for background tasks because any break-points in the code DO NOT cause the task to stop and wait for the user to debug. To illustrate the process of debugging we will use background task TS99500020 (ZBUS2105-ZNOTIFYREQRELEASE) that sends a notification (not a work item) to a recipient when the requisition is released has been added to the sample workflow template.

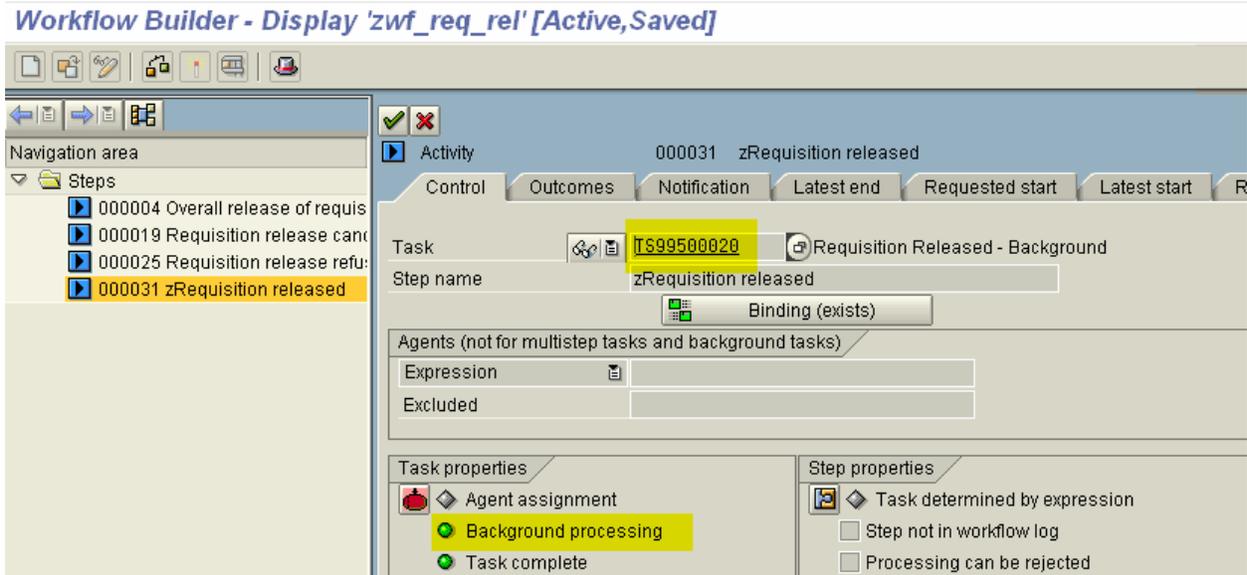
Hint: Several options are available for creating notifications via workflow. The SAP delivered 'SendMail' task, creating a custom method (as illustrated in this example with TS99500020/zmm_req_ok standard task), or creating a simple method w/ begin_method and end_method (adding text to task description and using report RSWUWFML to generate

notifications for the work item). To incorporate several different notifications into a workflow process it is sometimes simpler NOT to use the standard SAP 'SendMail' task and create your own method for the process, e.g. it's easier to dynamically create your agents for the notification from a custom method than using the standard 'SendMail'.



To debug, use the workflow builder (transaction PFTC > enter 'Workflow template' for task type > enter task number '99500034' for Name > select display > click on the 'Workflow builder' button) to call up the

workflow and double click on the background task TS99500020 (step '00031') in the process flow diagram to launch the workflow template task detail screen.



Next, double click on the task to launch the task details.

Standard Task: Display

Container Initial values	
Standard task	99500020 ZMM_REQ_OK
Name	Requisition Released - Background
Development class	ZFLO Appl. component BC-BMT-WF
Basic data Description Alternative methods Triggering events	
Name	
Abbr.	ZMM_REQ_OK
Name	Requisition Released - Background
Work item text	Purchase requisition & _WI_Object_Id.Number& released
Release status	Implemented
Object method	
Object type	ZBUS2105 zPurchaseReq
Method	ZNOTIFYREQRELEASE zNotifyRelease
	<input checked="" type="checkbox"/> Synchronous object method <input type="checkbox"/> Object method with dialog
Execution	
	<input checked="" type="checkbox"/> Background processing <input type="checkbox"/> Executable as form

Double click on method 'ZNOTIFYREQRELEASE' to launch the BOR definition of the method and then place the cursor on the 'ZNOTIFYREQRELEASE' method.

Display Object Type ZBUS2105

Object type ZBUS2105 SAPTips Purchase Requisition

- Interfaces
- Key fields
- Attributes
- Methods
 - zPurchaseRequisition.CreateFromData ✓ Create purchase requisition
 - zPurchaseRequisition.Change ✓ Change purchase requisition
 - zPurchaseRequisition.GetItems ✓ Read purchase requisition item
 - zPurchaseRequisition.GetItemsForRelease ✓ List purchase requisitions awaiting release
 - zPurchaseRequisition.Delete ✓ Delete/close purchase requisition
 - zPurchaseRequisition.GetDetail ✓ Display purchase requisition details
 - zPurchaseRequisition.GetReleaseInfo ✓ Detailed release information on purchase requisition
 - zPurchaseRequisition.Release ✓ Release purchase requisition
 - zPurchaseRequisition.ResetRelease ✓ Cancel release of purchase requisitions
 - zPurchaseRequisition.SingleRelease ✓ Individual release
 - zPurchaseRequisition.ExistenceCheck ✓ Check existence of object
 - zPurchaseRequisition.Display ✓ Display object
 - zPurchaseRequisition.InfoReleaseReset ✓ Info: Release cancelled
 - zPurchaseRequisition.Edit ✓ Process purchase requisition
 - zPurchaseRequisition.InfoReleaseRejected ✓ Info: Release refused
 - zPurchaseRequisition.InfoReleaseEffectuated ✓ Info: Release effected
 - zPurchaseRequisition.zNotifyReqRelease** zNotification of Requisition Release

Make sure the cursor is on the 'zNotifyReqRelease' method and click program. Insert a soft break-point in the code.

Object Type: Editor Display Program ZBUS2105


```

47 begin_method znotifyreqrelease changing container.
48
49 *****
50 * Data Declarations
51 *****
52 data: gw_purchase_req_nr like eban-banfn.
53 gw_purchase_req_nr = object-key-number.
54
55 data: gw_rcode   like syst-subrc.           "return code
56
57 * internal tables
58 data:
59     itab_objcont like solisti1 occurs 5 with header line, "notif.
60     itab_reclist like somlreci1 occurs 5 with header line, "receivers
61     result_tab like swhactor occurs 0 with header line.
62
63 * Structures
64 data:
65     gs_doc_chng      like sodocchgi1.
66
67 *****
68 *** Reciever List
69 *****
70 itab_reclist-receiver = OBJECT-_EBAN-ERNAM.   "Creator of Req
71 itab_reclist-rec_type = 'US'.
72 append itab_reclist.
73
74 *****
75 *** Content of Notification
76 *****
77 * refresh itab that stores the notificaiton text
78 refresh itab_objcont
79 * NAME of office notificaiton object
80 as doc chng-obi name = 'SAPTips: Workflow Debuadina'.

```

Green arrow out back to the BOR definitions and (while in CHANGE mode) DOUBLE CLICK ON 'zPurchaseRequisition.zNotifyReqRelease' to pull up the method attributes. WE WANT TO SET THE DIALOG BUTTON SO WORKFLOW DOES NOT AUTOMATICALLY PROCESS THE TASK (LATER WE WILL CHANGE THIS BACK, i.e. after any issues have been determined from the debug analysis)

The screenshot shows the configuration window for the method 'zNotifyReqRelease'. The window title is 'Method zNotifyReqRelease'. The configuration is as follows:

Method	zNotifyReqRelease
Object type	ZBUS2105
Release	46C
Status	implemented
Name	zNotifyRelease
Description	zNotification of Requisition Release

Below the table are three tabs: 'General', 'Result type', and 'ABAP'. The 'General' tab is selected and contains the following options:

- Dialog
- Synchronous
- Result parameter
- Instance-independent

At the bottom of the window are three icons: a green checkmark, a speech bubble, and a red X.

Save, generate, and then green arrow back to the task description. You will notice that the attributes of the task have changed. The 'Object method with dialog' flag has been set.

Standard Task: Display

Standard task	99500020 ZMM_REQ_OK
Name	Requisition Released - Background
Development class	ZFLO
Appl. component	BC-BMT-WF

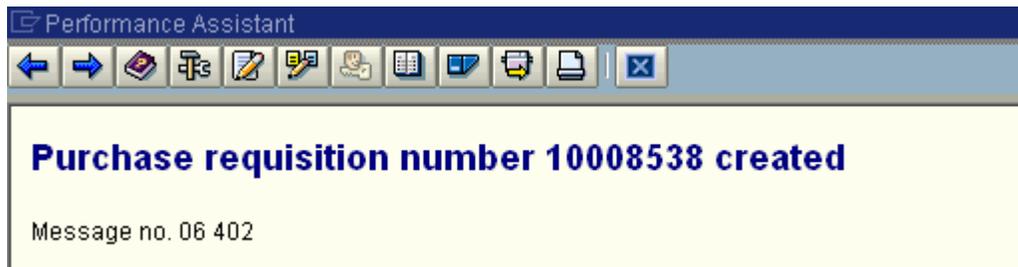
Basic data	Description	Alternative methods	Triggering events
------------	-------------	---------------------	-------------------

Name	
Abbr.	ZMM_REQ_OK
Name	Requisition Released - Background
Work item text	Purchase requisition & _WI_Object_Id.Number& released
Release status	Implemented

Object method	
Object type	ZBUS2105 zPurchaseReq
Method	ZNOTIFYREQRELEASE zNotifyRelease
<input checked="" type="checkbox"/> Synchronous object method <input checked="" type="checkbox"/> Object method with dialog	

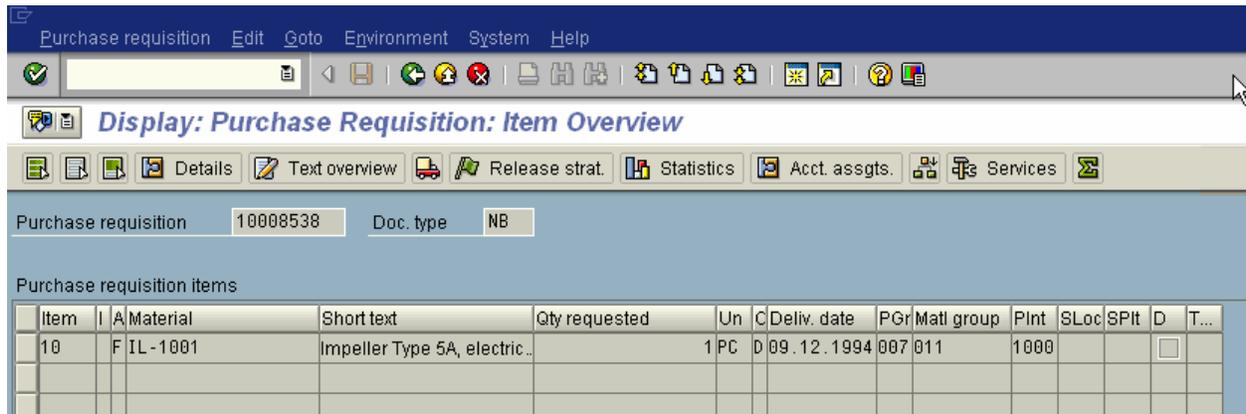
Execution	
<input type="checkbox"/> Background processing	<input type="checkbox"/> Executable as form
<input type="checkbox"/> Confirm end of processing	

Green arrow twice back to the workflow builder and regenerate your workflow. Let's create a new purchase requisition (subject to the release strategy). Workflow will now stop at this task and we will be able to execute it from the work item overview.

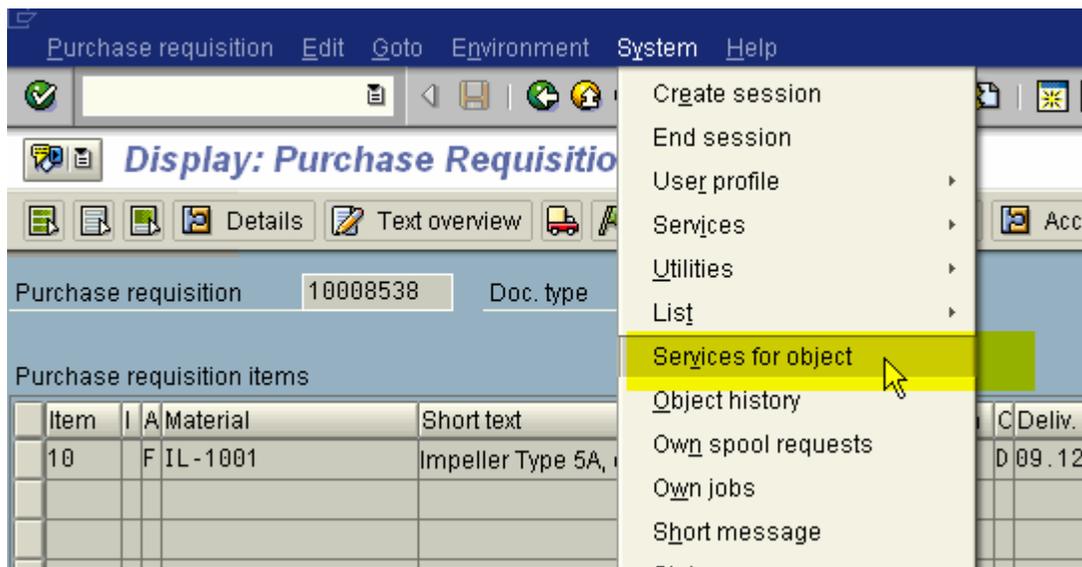


Let's use a helpful tool from SAP called Generic Object Services (GOS – available from release 4.0 onwards) to identify the workflow for the requisition we just created. GOS is activated by the application transactions and enhances the SAP functionality by allowing you to send email, add comments, review workflow history, and many other functions. After the requisition is created display it using transaction ME53.

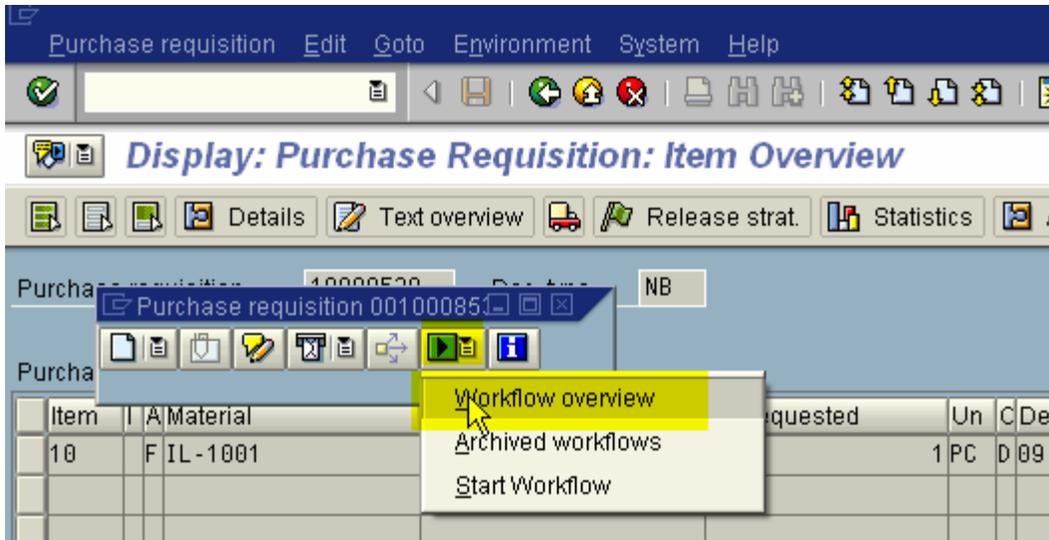
Note: as an alternative we can use SWI1 (work item overview to find the work item).



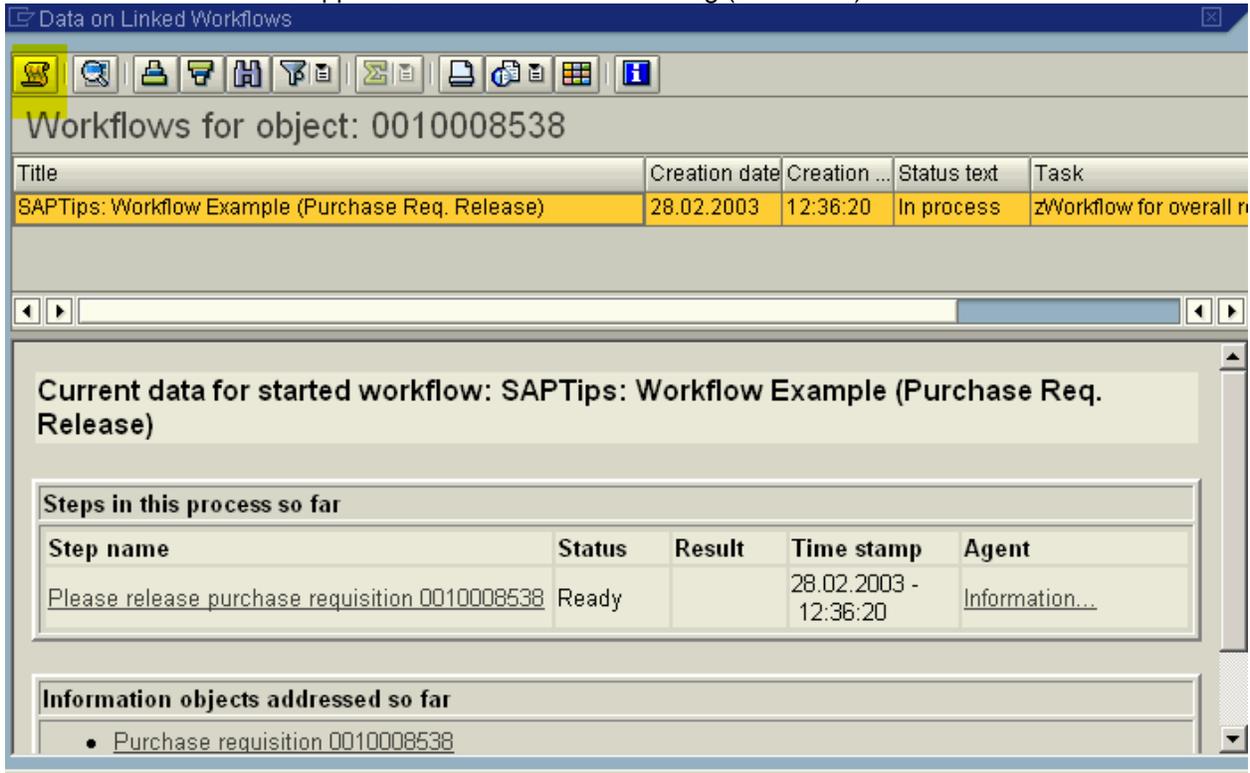
Use the menu and navigate to System > Services for Object



A new window appears with services > click on the workflow overview to display all workflows assigned to the purchase requisition.



Data on linked workflows appears > click on the workflow log (scroll icon)



The workflow log appears. Since the background task we have created occurs after the first dialog (overall release) task we must first execute the overall release. Click on work item ID 547778 (overall release task).

Workflow Log (View With Technical Details)

Agent Object Graphic Optimize width Choose Save ActiveX

Workflow zWorkflow for overall release of req.
 Workflow instance SAPTips: Workflow Example (Purchase Req. Release)
 Instance number 000000547777
 Start date 28.02.2003 Started by WF-BATCH
 Start time 12:36:20 Current status In process

View: Workflow chronicle

Error	St	ID	Node number	Task	Result
Error Agent			Executed	action	Date Time Object
		547777	1	SAPTips: Workflow Example (Purchase Req. Rel	% Workflow st
		547778	4	Please release purchase requisition 00100008538	

By clicking overall release work item 547778, the work item execution window appears. Execute the work item.

Note: if you cannot execute the work item from SWI1 then you are most likely not assigned as an agent to the task. The easiest way to resolve is to change the task attribute to 'General' as described previously in this article.

Please release purchase requisition 0010008538

Create Reserve Mail Log Graphic

Basic data Activities Available objects

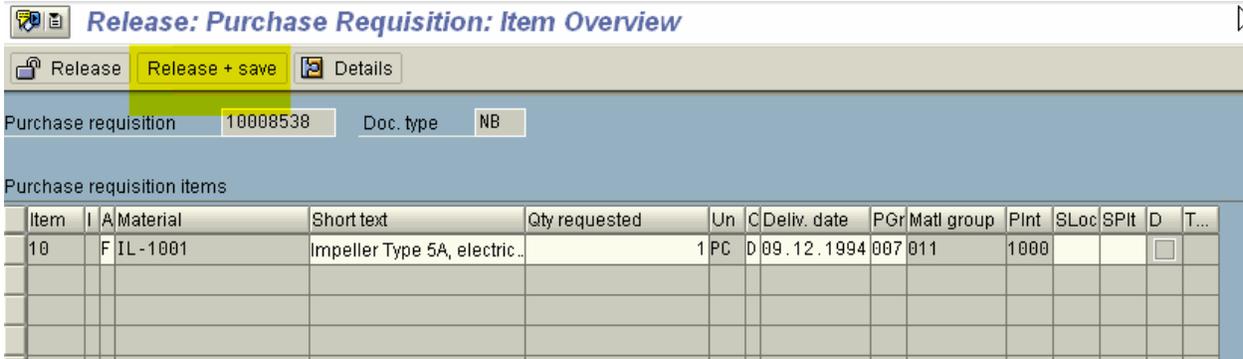
Please release purchase requisition 0010008538

Work item attributes

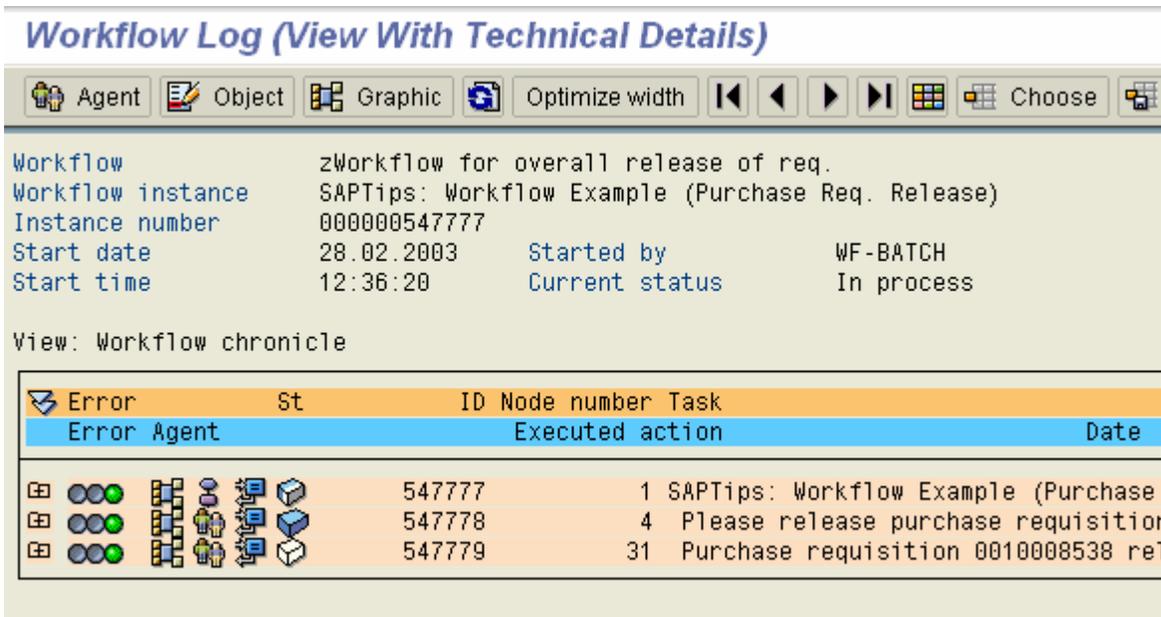
Start by Not defined
 End by Not defined
 Forwarded by
 Priority 5 Medium
 Status text Ready
 Creation date 28.02.2003 - 12:36:20
 Processed from 00.00.0000 - 00:00:00
 Open requests 0

Description **Attachments**

By executing the work item the task (method) calls transaction ME54 to release the req. Click 'Release + save' and release the requisition.



After the requisition is released you return to the work item > green arrow back to the workflow builder. You will notice that workflow now stopped on our background task (here it is work item 547779) BECAUSE WE SET IT TO A DIALOG TASK.



We can click work item id 547779 to launch the work item execution screen.

The screenshot shows the SAP Work Item Manager interface. At the top, there is a toolbar with icons for navigation and actions like 'Create', 'Reserve', 'Mail', 'Log', and 'Graphic'. Below the toolbar are three tabs: 'Basic data', 'Activities', and 'Available objects'. The main content area is titled 'Purchase requisition 0010008538 released'. Underneath, there is a section for 'Work item attributes' with a list of fields and their values. The 'Priority' field is currently set to '5 Medium' and has a 'Save changed priority' button next to it. Below the attributes is a 'Description' section with the text: 'SAPTips Workflow Example: Background Task for notification of purchase requisition release. This does NOT create a dialog work item.' To the right of the description, there is a partially visible 'Attachments' section.

Work item attributes	
Start by	Not defined
End by	Not defined
Forwarded by	
Priority	5 Medium <input type="button" value="Save changed priority"/>
Status text	Ready
Creation date	28.02.2003 - 12:53:30
Processed from	00.00.0000 - 00:00:00
Open requests	0

Description

SAPTips Workflow Example: Background Task for notification of purchase requisition release. This does NOT create a dialog work item.

Attachments

Execute the work item. You will now be able to debug the background method associated with the task. Technically, we have made the method dialog to debug it. After you have finished debugging the method be sure to change the attributes back to background (switch off the dialog indicator in SWO1) and then regenerate the workflow via transaction PFTC.

ABAP_DEBUGGER

The screenshot shows the ABAP Debugger interface for program ZBUS2105. The main program and source code of are both ZBUS2105. The code is as follows:

```

FORM ZNOTIFYREQRELEASE
    itab_reclist like somlreci1 occurs 5 with header line, "receivers
    result_tab like swhactor occurs 0 with header line.

    * Structures
    data:
        gs_doc_chng      like sodocchg1.

    *****
    *** Reciever List
    *****

    itab_reclist-receiver = OBJECT-_EBAN-ERNAM.  "Creator of Req
    itab_reclist-rec_type = 'US'.
    append itab_reclist.

    *****
    *** Content of Notification
    *****

    * refresh itab that stores the notificaiton text
    ►  refresh itab_objcont.
    * NAME of office notificaiton object
    gs_doc_chng-obj_name = 'SAPTips: Workflow Debugging'.
    * SUBJECT of email office notificaiton
    clear gs_doc_chng-obj_descr.
    gs_doc_chng-obj_descr = 'SAPTips: Purchase Req Released'.
    * TEXT of Notification
    *****
  
```

Summary of Trouble Shooting Transactions

Once you are able to initiate the debugging sessions all the standard ABAP debugging principles apply. Here's a review of some important transactions to assist in troubleshooting your workflows.

#	Transaction	Description	Notes
1.	SWI1	Workitem overview	This is the most used transaction to display

			workflows created in the SAP systems. Use the technical workflow log to analyze single step tasks in a multi-step workflow. Review the container element parameter values passed in to each task, change container element parameters to test different variations of the workflow, and determine who the work items are assigned to (agent resolution).												
2.	PFTC	This is the general task maintenance transaction used to develop multi-step and single step tasks. This is the entrance point to the workflow builder process diagram.	<p>Later versions of SAP use the following transactions:</p> <table border="0"> <tr> <td>PFTC_CHG</td> <td>Change Tasks</td> </tr> <tr> <td>PFTC_COP</td> <td>Copy Tasks</td> </tr> <tr> <td>PFTC_DEL</td> <td>Delete Tasks</td> </tr> <tr> <td>PFTC_DIS</td> <td>Display Tasks</td> </tr> <tr> <td>PFTC_INS</td> <td>Create Tasks</td> </tr> <tr> <td>PFTC_STR</td> <td>Task Maintenance -></td> </tr> </table> <p>Be sure to request authorization to PFTC since it is less cumbersome to use than the individual transactions.</p> <p>In single-step tasks use the menu and go to 'Additional Data' > 'Agent Assignment' to check the task attributes. Remember 'General' tasks can be executed by anybody.</p>	PFTC_CHG	Change Tasks	PFTC_COP	Copy Tasks	PFTC_DEL	Delete Tasks	PFTC_DIS	Display Tasks	PFTC_INS	Create Tasks	PFTC_STR	Task Maintenance ->
PFTC_CHG	Change Tasks														
PFTC_COP	Copy Tasks														
PFTC_DEL	Delete Tasks														
PFTC_DIS	Display Tasks														
PFTC_INS	Create Tasks														
PFTC_STR	Task Maintenance ->														
3.	SWU3	The area menu IMG for setting up the workflow engine.	<p>Several steps are required to set-up the engine. Errors usually occur w/ the RFC destination (see transaction SWUB) below.</p> <p>Use the 'Start verification workflow' to validate the workflow engine is set-up correctly.</p>												
4.	SWUB	Workflow RFC destination	<p>Synchronize the password and use the test button to make sure the RFC destination is working correctly.</p> <p>Often errors exist because someone changed the WF-BATCH user id password.</p>												
5.	SM21	System Log	If you cannot figure out why a particular workflow is NOT working always check the system log.												
6.	ST22	Short Dump Log	If you cannot figure out why a particular workflow is NOT working always check for short dumps.												

7.	SWE2	Event Linkage Table	If you are expecting workflows and not receiving any make sure the event is linked to a workflow receiver.
8.	SWELS	Turn on/off the event trace log	<p>This can help resolving event issues. Validate, at minimum, that an event is being created and then check SWE2 for the event linkage. If no event is being created then there's most likely a problem with the activation of events in the application configuration or other areas.</p> <p>IMPORTANT: IN PRODUCTION YOU SHOULD TURN OFF THE EVENT LOG FOR DAILY USE AND ONLY TURN ON WHEN TROUBLESHOOTING. If the log is on it has a tendency to fill up the log tables and cause errors if the system is not properly maintained.</p>
9.	SWEL	Display Event Trace	Display any events you expected to be triggered.
10.	SWU_OBUF	Synchronize run time buffer	As of 46 SAP includes a task and org assignment buffer for increased performance. Any time org assignments to tasks have been made BE SURE TO SYNC THE BUFFERS . Otherwise, new agent assignment rules may not be taken into considerations, i.e. you may have fixed an issue but the buffer is using the old data!
11.	SWUS	The workflow test tool.	<p>This is very helpful to start workflows for transactions where no workflows were created but were supposed to.</p> <p>It's also a GREAT tool to start debugging tasks after you inserted a break-point.</p> <p>Review other work items for similar tasks via transaction SWI1 (display the container elements) if you need help identifying values for the 'Input Data'.</p>
	SWUE	Workflow event test tool.	Similar to the SWUS but this creates the event. Again, use SWI1 to identify container elements (event > ws).
12.	SM58	tRFC buffer errors	<p>If any workflow engine issues exist the workflow will log errors in the tRFC event tables.</p> <p>The use of SM21, ST22, and SM58 are</p>

			essential in identifying workflow issues.
13.	SCOT	SAP Connect Configuration	Any notifications (not work items) will be logged here. If there are issues with end users receiving notifications review SCOT. Most often, the issue is that program RSCONN01 is not running to send notification out to the SENDMAIL –or- exchange server.
14.	SWLD	Old area menu for workflow development	Exit out of the easy menu (/n), then enter /nSWLD to get to the old area menu.
15.	SWPC	Re-start workflows after system crash	
16.	SWIA	Execute work item w/o agent check	This is a great transaction to bypass agent resolution issues and focus on other problems.
17.	SWPR	Re-start workflow after error.	