An expert guide to new SAP workflow capabilities
Introduction

Handling Data with Containers and Expressions

Controlling Flow with Conditions and Blocks

Dealing With Events

Distributing Work

Monitoring Your Processes

Testing Workflows and Using ABAP Objects
What This Session Is About

This session…

… describes new features in the SAP Business Workflow area
  ■ Web Application Server releases 620 to NetWeaver 2004S

… is not an introduction to SAP Business Workflow
  ■ You have to be familiar with SAP Business Workflow to fully profit from this session

… covers a lot of different independent topics
  ■ It should give you an overview and a first impression of the new concepts
  ■ More details can be found
    ◆ in the slide notes
    ◆ in the appendix
    ◆ in the documentation

… is about technology, not about real-world scenarios
New workflow features introduced in this presentation are associated with the Web Application Server release they are first available in.

The release number is highlighted in the upper right corner of each slide.
Introduction

Handling Data with Containers and Expressions

Controlling Flow with Conditions and Blocks

Dealing With Events

Distributing Work

Monitoring Your Processes

Testing Workflows and Using ABAP Objects
Re-implementation of the Container

Complete reimplementation of container technology

No restrictions concerning data types
- DDIC types with arbitrary lengths
- Binary data
- Structures with arbitrary depth and complexity
- ABAP OO references

Container is by default stored as XML document
- Facilitates diversity of supported data types
- Saves storage space (especially for large containers/structures)
  - No database entry for each structure field required anymore
- Entails serialisation / deserialisation effort

Old container persistence (SWCONT, SWCONTOB) still supported
- Compatibility mode
- Workflow Builder setting (Appendix)
The old container was represented as an internal table

- Manipulating the container meant using awkward container macros

The new container implementation is ABAP Objects based

A (released) API is provided for reading and manipulating container data

- the ABAP Objects interface IF_SWF_IFS_PARAMETER_CONTAINER

This interface represents the container already in a number of workflow APIs (and will get more ubiquitous from release to release)

IF_SWF_IFS_PARAMETER_CONTAINER contains the following methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEAR</td>
<td>Resets an element to its type-based initial value</td>
</tr>
<tr>
<td>GET</td>
<td>Delivers the value of an element</td>
</tr>
<tr>
<td>GET_TYPE</td>
<td>Delivers the data type of an element</td>
</tr>
<tr>
<td>GET_VALUE_REF</td>
<td>Delivers a reference to the value of an element</td>
</tr>
<tr>
<td>LIST_NAMES</td>
<td>Delivers a list of names of all elements</td>
</tr>
<tr>
<td>SET</td>
<td>Sets the value of an element</td>
</tr>
</tbody>
</table>
Expressions

Expressions now provide index access to multi-line elements

<table>
<thead>
<tr>
<th>&amp;table[index]&amp;</th>
<th>Access to an entire table line</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;table[index].columnName&amp;</td>
<td>Access to a component of a table line</td>
</tr>
<tr>
<td>&amp;table[].columnName&amp;</td>
<td>Projection to a single column</td>
</tr>
<tr>
<td>&amp;customers[1].orders[2]&amp;</td>
<td>Second order of first customer</td>
</tr>
</tbody>
</table>

Expressions can contain functional method calls to BOR or ABAP Objects

- The expression evaluates to the result value of the method
- It is possible to pass parameters
- Methods used in expressions must not have side effects
  - „Read only“ methods
  - No database changes

| Instance method w/o parameters | &my_object.get_value( )& |
| Instance method w parameters   | &my_object.methodA( param1=&exp1&; param2=17 )& |
| Static method                  | %my_class.static_method( param1=&exp2& )% |

No more restrictions concerning data types

- ABAP-OO object references
- Strings
- Nested structures
New Binding Editor

6.20

source container

drag & drop
target container

drag & drop
import binding

drag & drop
export binding

binding instructions
Binding Instructions

Source and Target of a binding instruction are expressions
- Assigning a table line to a structure component is now possible (if compatible)

Several binding operators are available
- for expression to expression binding (as above)
- for container to container binding

The button lets you choose the right binding operator

<table>
<thead>
<tr>
<th>Expression to expression</th>
<th>Container to container</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign value</td>
<td>Merge (Copy all elements)</td>
</tr>
<tr>
<td>Append to table</td>
<td>Programmed Binding (Container-IF)</td>
</tr>
<tr>
<td>Initialize</td>
<td>Programmed Binding (SWCONT)</td>
</tr>
<tr>
<td>XSLT Transformation</td>
<td></td>
</tr>
</tbody>
</table>

→ see Appendix for further configuration possibilities
Introduction
Handling Data with Containers and Expressions
Controlling Flow with Conditions and Blocks
Dealing With Events
Distributing Work
Monitoring Your Processes
Testing Workflows and Using ABAP Objects
Create Work Item Condition

Instantiation

READY
SELECTED
STARTED

CHECKED

Execution

Complete Execution Condition

Complete Work Item Condition

COMPLETED

see Appendix for further details
SWITCH: The New Multiple Condition Flavor

1. Double-click on multiple condition icon

Which cost center interval?

2. Select Multiple Condition Flavour

3. Enter Branches and Conditions

4. Name Default Branch

- Each branch of the SWITCH carries a condition definition
- The conditions are evaluated in the given order
- The first branch whose condition evaluates to TRUE is taken
- If none of the conditions evaluates to TRUE, a default branch is taken

Condition 1 is TRUE
Condition 2 is TRUE
Default branch
Blocks... 

- are modeling elements 
- can contain other modeling elements 
- represent a data sub-context 
  → all modeling elements within the block refer to this data context (binding, conditions, …) 
- have a data interface 
- have one start and one end node 
- are represented by block work items 
  → new work item type 
- can be deadline monitored (latest end only) 
- can catch exceptions (see Appendix)
Blocks: Work Item Hierarchy

- Block items introduce new levels in work item hierarchies
  - Blocks can contain dependent work items

- Blocks are represented as folders in the workflow log

- Block structure is sustained at runtime
  - Block is completed only if dependent work items are in a final state
  - If a block is cancelled, all its dependent work items are cancelled, too
- Every block has a container
- Nested blocks form nested data contexts
  - Parent relationships between containers
  - Workflow is the top level block
- Parent context is visible within sub context
- Bindings can be defined from
  - the parent context to the sub context
    (Import binding at block instantiation)
  - the sub context to the parent context
    (Export binding after block completion)
Block instantiation can be dynamically controlled by multiline container elements (ParForEach)

- Similar to the „ParForEach“ dynamic parallel processing for activities!

For each entry in the multiline element, one block instance is created
- Corresponding line data transported to the block context via binding

Process continues after the ParForEach block if either
- all instantiated parallel blocks are completed or
- a special join condition evaluates to TRUE after completion of a block
Block execution can be dynamically controlled by multiline container elements (ForEach)

For each entry in the multiline element, the sequence of steps defined within the block is executed once

- Only one block item is created!
- ForEach loop semantics

Process continues after the ForEach block if either

- all table entries are processed or
- a special condition evaluates to TRUE after completion of a loop sequence

Block definition

© SAP AG 2006, 19
Local workflows...

- are control flow snippets
  - not part of the main process sequence
- are just special blocks
  - represented by block work items
- are started via local events
  - data binding event – local flow
- have full access to process data
- can be instantiated an arbitrary number of times
- are cancelled if still active when main process completes
Dealing With Events
The workflow instance can listen to events
- Event couplings only exist if the corresponding object exists in the workflow container
- The receiver type tells you how the workflow processes the event

<table>
<thead>
<tr>
<th>Receiver Type</th>
<th>Reaction of Workflow to Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel Workflow</td>
<td>■ The workflow instance is set to status CANCELLED</td>
</tr>
<tr>
<td>Cancel and Restart Workflow</td>
<td>■ The workflow instance is set to status CANCELLED&lt;br&gt;■ A new instance is started with the same data</td>
</tr>
<tr>
<td>Evaluate Pre- and Postconditions</td>
<td>■ Conditions „Create Work Item“ and „Complete Work Item“ are evaluated</td>
</tr>
<tr>
<td>Reevaluate Rules</td>
<td>■ Agent determination of all active dialog work items is repeated</td>
</tr>
</tbody>
</table>
**Workflow Header Events (2)**

- WAS 640 (NW04) offers you two new receiver types

<table>
<thead>
<tr>
<th>Receiver Type</th>
<th>Reaction of Workflow to Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait Step Event</td>
<td>▶ Event is delivered to existing appropriate wait step (event work item)</td>
</tr>
<tr>
<td></td>
<td>▶ If no active event work item exists, the event is parked</td>
</tr>
<tr>
<td></td>
<td>➔ see one of the next slides for parked events and the extended wait step!</td>
</tr>
<tr>
<td>Trigger Local Event</td>
<td>▶ A specified local event is triggered</td>
</tr>
<tr>
<td></td>
<td>▶ This local event could</td>
</tr>
<tr>
<td></td>
<td>◆ start local workflows</td>
</tr>
<tr>
<td></td>
<td>◆ complete wait steps (event items)</td>
</tr>
<tr>
<td></td>
<td>➔ see one of the next slides for the extended wait step!</td>
</tr>
</tbody>
</table>
Problem/Feature 1:
- Workflow can only react on events if appropriate wait step has already been instantiated.
- If event is raised "too early", it is lost for the process.

Problem/Feature 2:
- Events always complete all corresponding active wait step instances (event items).
The process instance can act as an intermediate event storage

- If active receiver event items exists, the one created first receives the event
- If no active receiver event item exists, the event is parked
- The first matching event item created consumes the first event parked
- In any case: One event is delivered to exactly one event item

➔ See Appendix for further details
Event couplings are created using
- event name (design time information)
- event object instance (run time information)

The object instance has to be known to the process

**Problem:** How to wait for events of yet unknown objects?

Knowing an object means knowing its key (unique identifier)

But: objects can also be identified using semantic information

**Goal:** Wait for events of objects semantically coupled to objects known
Correlations are objects which group other objects by semantic criteria (correlation key)

⇒ “objects having to do with each other”

Event couplings can also be created using

- event name (design time information)
- event object type (design time information)
- correlation instance (run time information)

The object instance does not have to be known to the process

⇒ but at least one correlated object instance has to

⇒ Example: know order to identify correlated invoice
Correlations are instantiated using one participating object instance

- (semantic) correlation key has to be determined

   ➜ Example: order ID

**Wait step (event item) is instantiated using**

- correlation instance (semantic key)
- object type
- event

   ➜ Example: wait for creation of invoice referring to known order

Event of unknown, yet correlated object is delivered to event item via the correlation instance

➜ see Appendix for further details
Wait Step: Four Different Flavours

Wait for event
- Conventional wait step
- Wait for global event
- Correlations can be used

Wait for condition
- Condition specified in wait step
- Evaluated by periodic background job SWWCOND

Wait for event using workflow
- Wait for global event
  ◆ received by workflow instance
  ◆ dispatched to event item
- Correlations can be used

Wait for local event
- Wait for local event raised by trigger step within the same workflow instance
Introduction
Handling Data with Containers and Expressions
Controlling Flow with Conditions and Blocks
Dealing With Events
Distributing Work
Monitoring Your Processes
Testing Workflows and Using ABAP Objects
Means to improve SBWP performance have been provided.

Scenarios that benefit most are:

- Many agents work on large pools of work items (e.g. call center scenario)
  - Large, slow work lists
  - Necessity to often refresh the work list (due to work item „overlaps“)

- Extensive usage of dynamic columns
  - Particularly with large work lists

- Grouping of work items in large work lists using container data
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key

To mitigate SBWP performance issues, BAdIs have been defined.
SBWP-BAdI: Filtering Work Lists

- **BAdI WF_BWP_SELECT_FILTER** enables the reduction of the number of work items displayed in the Business Workplace workflow inbox
  - It imports the entire work list (work item header information)
  - It exports the (reduced) work list to be displayed as inbox

- Custom filtering algorithms can be implemented
  - Example implementation based on random numbers available

- Performance critical operations applied to reduced work list
  - Default attributes, dynamic columns, work item text in different languages, ...

- Availability: WAS 620 SP 44, WAS 640 SP 9 (see note 765783)
Dynamic columns can be customized on task level
   • Display work item container information in Business Workplace inbox

Usage of dynamic columns is performance critical
   • Read container for every single work item
   • Instantiate objects and read object attributes
   • Container and Business Object Repository are not mass-enabled

BAdI WF_BWP_DYN_COLUMN can help to improve performance

The BAdI implementation facilitates
   • custom buffering
   • mass selection from database

Availability: Future Support Packages for WAS 620 and WAS 640
   • see note 848382 (not released yet)
**SBWP-BAdI: Evaluating Default Attributes**

- Default object attributes are displayed as object representatives
- Determination of object attributes is time consuming
  - Not mass-enabled
  - Very critical for large numbers of work items
- BAdI WF_BWP_OBJ_ATTRIBUTE can help to improve performance
  - Allows a custom, mass-enabled determination of default attributes
- Availability: Future Support Packages for WAS 620 and WAS 640
  - see note 848382 (not released yet)
Email Notifications

Replacement for SAP MAPI is provided
- Client-based MS Outlook integration of SAP Business Workflow
- Support ends 10/2005

New solutions
- Server-based (zero footprint)
- Mail client independent
- Covers 90% of workflow-related MAPI customer scenarios

Different replacements for different SAP releases

Report RSWUWFML2
- Polling report sending email notifications
- Releases 4.6C – 620

Extended Notifications
- Server notification framework
- Releases > 640
Extended Notifications: Overview

SAP Business Workflow

collect workitems

process workitem

End user

send e-mails

Email client

SAP GUI or Portal UWL

Extended Notifications

Email

© SAP AG 2006
Notify users about work items that need to be processed
Send workitems to groupware (e.g. MS Outlook or Lotus Notes) carrying direct callbacks to backend transactions

- Execution of web-based work items (Web Dynpro, People centric UI, …)
- Execution of work items via SAP GUI for Windows

First step in workflow verification

Choose one of the decision options given. This completes the processing of this step.

Before you make a decision, you can display the attachments and objects which have been attached to the user decision. You can also create your own attachments.

If you choose Cancel, the user decision remains in your inbox for processing.

- Execute background step immediately
- Execute background step in a minute

Display Work Item
Execute Work Item

If you have problems when logging on to the system, contact your system administrator.
The Workflow Information System (WIS) has become obsolete with WAS release 610.

Customers do their reporting using BI.

Workflow projects with BI requirements:
- could not use standard workflow-related data extraction
- had to implement their own extraction logic for workflow data

This gap has now been closed:
- Technical work item data is available in BI as infocube
  - How many workflows of type ‘xy’ were executed?
  - What is the average processing time for a given workflow (step)?
  - How many steps of type ‘xy’ were executed by org unit ‘abc’?

Workflow and Business Objects worlds are linked by work item to object relation information available.
BW Integration for SAP Business Workflow (2)

0BPM_C01_Q0001: BPM Reference Query

0BPM_C01: Aggregated BPM Data

0BPM_DS01: Work items
0BPM_DS02: Logs
0BPM_DS03: Work item object relation
0BPM_DS04: Deadlines

0BPM_WIHEAD: Workitems
0BPM_LOGHIST: Logs
0BPM_OBJREL: Work item object relation
0BPM_DEADLINES: Deadlines

© SAP AG 2006, 40
Work item exit classes have to implement the ABAP Objects interface IF_SWF_IFS_WORKITEM_EXIT

Exit classes can be attached to
- Steps (activity, decision, wait, web activity) ➔ corresponding work items
- Blocks ➔ block items
- Workflows ➔ flow items

Exit classes are called at particular times in the work item lifecycle
The workflow runtime provides the work item exit with
- the name of the lifecycle event
- a work item context object (reference to IF_WAPI_WORKITEM_CONTEXT)

The work item context provides methods to
- read work item data (header, task, agents, texts, ...)
- get container handles (IF_SWF_IFS_PARAMETER_CONTAINER)
  - of the work item itself
  - of the super ordinate workflow
- read properties (see next slide)
- write messages to the workflow log
Properties are custom labels for workflow definition entities
  ◆ steps (activities, decisions, wait steps, web activities)
  ◆ blocks
  ◆ workflows

Properties can be used to
  ◆ categorize steps
  ◆ mark process states (milestones)

Properties are just name value pairs

They can be queried in work item exits (custom process reporting)
  ◆ „Is the current work item an approval work item?“
  ◆ „Is the current process still in negotiation phase?“
Application Specific Workflow Reporting

Workflow definition

- workflow started
- activity
- step property
- activity
- workflow completed

step with work item exit

Reporting tool

Work item exit

Application specific reporting database

Work item context
Introduction
Handling Data with Containers and Expressions
Controlling Flow with Conditions and Blocks
Dealing With Events
Distributing Work
Monitoring Your Processes
Testing Workflows and Using ABAP Objects
Start options can be specified when testing workflows (SWUS)

- Options enable better error analysis
  - Debugging
  - Tracing

<table>
<thead>
<tr>
<th>Option</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Workflow Asynchronously</td>
<td>The workflow instance is started via aRFC</td>
</tr>
<tr>
<td>Simulate background processing</td>
<td>Background steps are executed synchronously, not via tRFC</td>
</tr>
<tr>
<td></td>
<td>Valuable for debugging background steps</td>
</tr>
<tr>
<td>Start Dialog Automatically</td>
<td>Switch off synchronous dialog chains</td>
</tr>
<tr>
<td>Trace Level</td>
<td>Switch on workflow trace and set trace level</td>
</tr>
<tr>
<td>Debug Mode</td>
<td>Use aRFC instead of tRFC</td>
</tr>
</tbody>
</table>
Extended start options with release 700

<table>
<thead>
<tr>
<th>Option</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repeated Starts</td>
<td>Instantiate and start workflow a specified number of times</td>
</tr>
</tbody>
</table>
| Persistence Profile | Specify container persistence profile for single step tasks (TS...)  
|                   | The persistence profile for workflows (WS...) is specified in the workflow definition |
SAP Business Workflow now supports two object repositories
- BOR (Business Object Repository)
- SE24 (ABAP Objects)

Persistent object keys now have three components
- Object category (CATID) (BOR or ABAP Objects)
- Object type (TYPEID) (BOR object type or ABAP Objects classname)
- Object key (INSTID)

Both repositories equally well supported
- But: No delegation feature in SE24

SE24 support covers the same aspects as BOR support
- Container
- Expressions
- Methods
- Events
- Container can reference ABAP OO objects

- ABAP OO objects can be used in expressions
  - just as BOR objects are used

- ABAP OO object methods can be used in standard task definitions
  - just as BOR object methods are used
Resources

Public Web:

www.sap.com
http://service.sap.com/bpms


Related SAP Education Training Opportunities

http://www.sap.com/education/
BIT600, BIT601, BIT603, BIT610

Related Lectures

What you need to know about SAP NetWeaver Exchange Infrastructure message processing, monitoring, and troubleshooting
(Tuesday, March 28, 10:00 am - 11:15 am)
1. Start to use BI cubes for workflow reporting and for process analysis; this is a building block for Business Activity Monitoring.
2. Local workflows enable you to have a process within a process that can run as many times as the workflow requires.
3. Local events enable workflow flexibility without additional performance overhead.
4. Extended notifications replace SAPMAPI for sending work items to Outlook.
5. Switch your workflow container to use XML instead of SWCONT. This saves storage space since there is no database entry for each structure field required any longer.
6. Try to get involved in SAP NetWeaver Exchange Infrastructure. Many new workflow features are created to support the BPM runtime. Business Process Management (BPM) is a natural skill-extension to SAP Business Workflow.
7. Move towards the Universal Worklist as the inbox for workflow items.
Questions?

Q&A
Feedback

Please complete your session evaluation.

Be courteous — deposit your trash, and do not take the handouts for the following session.

Thank You!
Demo