

# BOPF Validations

SAP AG, 2012

# Disclaimer

---

This presentation outlines our general product direction and should not be relied on in making a purchase decision. This presentation is not subject to your license agreement or any other agreement with SAP. SAP has no obligation to pursue any course of business outlined in this presentation or to develop or release any functionality mentioned in this presentation. This presentation and SAP's strategy and possible future developments are subject to change and may be changed by SAP at any time for any reason without notice. This document is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. SAP assumes no responsibility for errors or omissions in this document, except if such damages were caused by SAP intentionally or grossly negligent.

# Agenda

---

Introduction

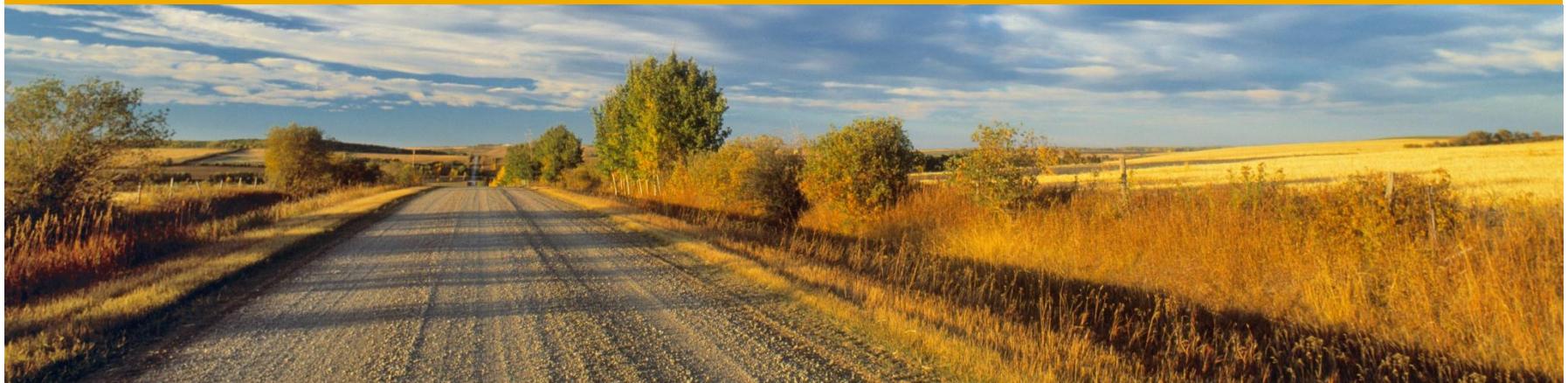
Action Validations

Implementation

Consistency Validations

Consistency Groups

Overview



# Introduction

# Definition

---

Validations are business logic entities that are triggered in certain situations to check different aspects of a given set of node instances.

Validations never modify any data!

Validations return messages and the keys of failed node instances.

There are two categories of validations:

- **Action validations**

Check whether necessary preconditions for executing an action on a node are fulfilled.

- **Consistency validations**

Check the consistency of a node instance



# Action Validations

# Action Validations

The screenshot shows an SAP ERP invoice document. At the top left is the SAP logo and the text "SAP AG, Neurostrasse 16, 69190 Walldorf". A red rectangular box highlights a question mark icon in the top right corner of the header area. In the header, there is sample data: *Invoice No.: 2005-008546*, *Customer No.: 15263741*, *Tax No.: 12 345 67890*, and *Date: 07/18/2005*. The main section is titled "Invoice" and contains the message "Your order 4711 of 07/12/2005". Below this is a table of items:

Item	Quantity	Art.-No.	Description	Price/U	Amount
1	1.00 PC	56452375	Sailing Boat	\$ 790.00	\$ 790.00
2	2.00 PC	75937255	Video Games	\$ 150.00	\$ 300.00
3	5.00 KG	43553452	Comics	\$ 100.00	\$ 500.00

Below the table are summary lines: "Subtotal \$ 1.590.00", "plus Shipping Costs \$ 8.00", "plus Packing Costs \$ 2.00", "Value \$ 1.600.00", "Tax (16%) on \$ 1.600.00 \$ 256.00", and "Total Value \$ 1856.00". To the right of the document, a large grey circle is partially visible, intersected by a thick red "X" and a dashed red "ISSUED" diagonal line.

## Motivation for action validations

- Only invoices with a specified billing party are to be issued.  
Thus we need to check whether the corresponding BOPF action `INVOICE_ISSUED` is currently allowed.
- Example:** Unless the complete customer address data is specified, the invoice cannot be issued.

# Action Validations

---

Action validations check whether it is possible to execute a certain action on a set of node instances.

Action validations only return messages and the keys of failed instances (i.e. instances for which the action may not be executed).

Action validations are triggered by attempts to execute an action. Therefore it is not necessary to configure request nodes.

The BOPF framework actions (e.g. “create”, “delete”, and “update”) can also be subject to an action validation (e.g. action validations can prevent instances with incomplete data from being created).

# Action Validations

## How to Create Them

The screenshot shows the SAP Business Object maintenance interface for the business object **ZCI\_CUSTOMER\_INVOICE**. The left pane displays the **Business Object Detail Browser** with the node structure. The **ROOT** node is selected, and a context menu is open over the **Validations** folder. The menu path **Create Validation > Action Validation** is highlighted. The right pane contains the configuration details for the new validation node, including the **Node** tab with fields for **Node Name** (ROOT) and **Description** (Root), and the **Node Settings** tab where **Node Type** is set to **Standard Node**.

**Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version**

**Business Object Detail Browser**

- ZCI\_CUSTOMER\_INVOICE** Customer Invoice
- Node Structure**
- Node Elements**
  - ITEM** Line Item
  - ROOT** Root
    - Node Categories**
    - Associations**
    - Determinations**
    - Validations**
      - Create Validation** **Action Validation**
      - Consistency Validation**
      - Help**
      - Alternative keys**
      - Status Variables**
      - Status Derivators**
      - Status Schemas**
      - Attribute Value Sets**
      - Authorization Field Mapping**
    - Groups**

**Node** Persistence Attribute Mapping Property Change Trigger

**Node**

**Node Name**: ROOT  
**Description**: Root

**Node Settings**

**Node Type**: Standard Node  
**Standard Node Cat.**: ROOT

**Data Model**

**Combined Structure**: ZCI\_S\_CI\_ROOT  
**Combined Table Type**: ZCI\_I\_CI\_ROOT  
**Data Structure**: ZCI\_S\_CI\_ROOT\_D  
**Data Structure (tr.)**:

# Action Validation Configuration

Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version

Validation Name	CHECK_ITEM_CURRENCY_CODE
Description	
Validation Settings	
Node	ITEM
Validation Cat.	Action Check
Class/Interface	ZCL_CI_V_CHECK_CURRENCY_CODE
<input checked="" type="checkbox"/> Check method implemented	
<input checked="" type="checkbox"/> Check Delta method implemented	
<input type="checkbox"/> Do Not Execute If Errors Have Occurred	
Test Data	
Test Data Container	
Variant	
Administrative Data	
Author	
Creation Date	
Last Changed By	
Changed On	

Buttons: ✓ ✗ ⏪ ⏩ ⌂

- Right-click on the validation to start the guided procedure.
- Enter a validation name and the description, and create a class using forward navigation.

# Action Validation Configuration

The screenshot shows two overlapping SAP dialog boxes. The top box is titled 'Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version' and contains a tree view of validation nodes under 'Validation Configuration (Node Cat.)'. The bottom box is also titled 'Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version' and contains a table for defining the 'Validation Sequence'.

**Validation Configuration (Node Cat.)**

Validation Configuration (Node Cat.)	Description
✓ CHECK_ITEM_CURRENCY_CODE	
✓ ROOT	
• <input type="checkbox"/> LOCK_ROOT	
• <input type="checkbox"/> UPDATE_ROOT	
• <input type="checkbox"/> DELETE_ROOT	
• <input checked="" type="checkbox"/> INVOICE_ISSUED	Invoice Issued
• <input type="checkbox"/> INVOICE_PAID	Invoice Paid
• <input type="checkbox"/> SAVE_ROOT	
• <input type="checkbox"/> UNLOCK_ROOT	
• <input type="checkbox"/> CREATE_ROOT	

**Validation Sequence**

Validation Sequence	Description
✓ CHECK_ITEM_CURRENCY_CODE	
• <input type="checkbox"/> Preceeding Validations	
• <input type="checkbox"/> Succeeding Validations	

- Choose the actions to be checked by the validation.
- Define the order of the action validations (not usually necessary) and press “Finish”.



# Implementation

# Implementation

## Overview

1. CHECK\_DELTA  
(optional)

Separation of  
Concerns

2. CHECK  
(optional)

3. EXECUTE

```
methods CHECK_DELTA
importing
  IS_CTX type /BOBF/S_FRW_CTX_VAL
  IO_READ type ref to /BOBF/IF_FRW_READ
changing
  CT_KEY type /BOBF/T_FRW_KEY
raising
  /BOBF/CX_FRW .
methods CHECK
importing
  IS_CTX type /BOBF/S_FRW_CTX_VAL
  IO_READ type ref to /BOBF/IF_FRW_READ
changing
  CT_KEY type /BOBF/T_FRW_KEY
raising
  /BOBF/CX_FRW .
methods EXECUTE
importing
  IS_CTX type /BOBF/S_FRW_CTX_VAL
  IT_KEY type /BOBF/T_FRW_KEY
  IO_READ type ref to /BOBF/IF_FRW_READ
exporting
  EO_MESSAGE type ref to /BOBF/IF_FRW_MESSAGE
  ET_FAILED_KEY type /BOBF/T_FRW_KEY
raising
  /BOBF/CX_FRW .
```



A validation is realized by a class implementing the `IF_FRW_VALIDATION` interface. A validation gets a table of keys of all relevant node instances. Using sequential execution of the interface methods, this table is reduced to a relevant set of instances.

# Implementation

## Method: CHECK\_DELTA

---

```
methods CHECK_DELTA
  importing
    IS_CTX          type /BOBF/S_FRW_CTX_VAL
    IO_READ         type ref to /BOBF/IF_FRW_READ
  changing
    CT_KEY          type /BOBF/T_FRW_KEY
  raising
    /BOBF/CX_FRW .
```

The CHECK\_DELTA method can be implemented to reduce the set of changed node instances (CT\_KEY) on which the validation is executed. This helps to improve performance.

- Filter out the node instances that were not changed in a way that is relevant for the validation.
- **Example:** A change made to the amount attribute does not make that instance relevant for a currency code validation.
- **Hint:** The method is only called for consistency validations (**exception:** action validations for CREATE, UPDATE and DELETE).

# Implementation

## Method: CHECK

```
methods CHECK
  importing
    IS_CTX      type /BOBF/S_FRW_CTX_VAL
    IO_READ     type ref to /BOBF/IF_FRW_READ
  changing
    CT_KEY      type /BOBF/T_FRW_KEY
  raising
    /BOBF/CX_FRW .
```

The `CHECK` method checks the relevance of the instances and removes irrelevant instances from `CT_KEY`.

- Unlike the `CHECK_DELTA` method, the semantics of instances is respected. Typically, the attributes of the instances are checked against special values in this method.
- **Example:** As long as the currency field is blank, there is no need to check the validity of the currency code.

# Implementation

## Method: EXECUTE

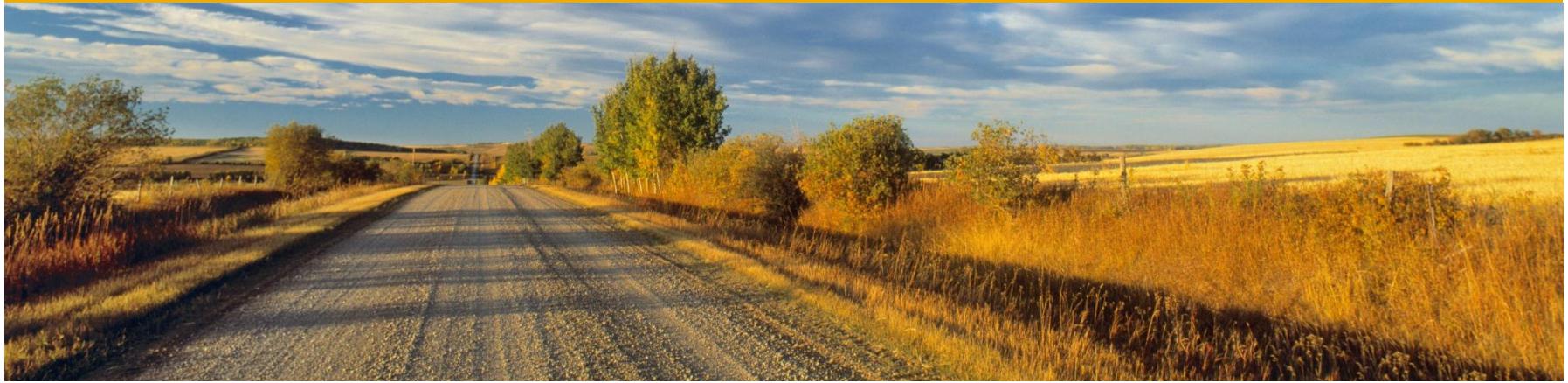
```
methods EXECUTE
  importing
    IS_CTX
    IT_KEY
    IO_READ
  exporting
    EO_MESSAGE
    ET_FAILED_KEY
  raising
    /BOBF/CX_FRW.
```

type /BOBF/S\_FRW\_CTX\_VAL  
type /BOBF/T\_FRW\_KEY  
type ref to /BOBF/IF\_FRW\_READ

type ref to /BOBF/IF\_FRW\_MESSAGE  
type /BOBF/T\_FRW\_KEY

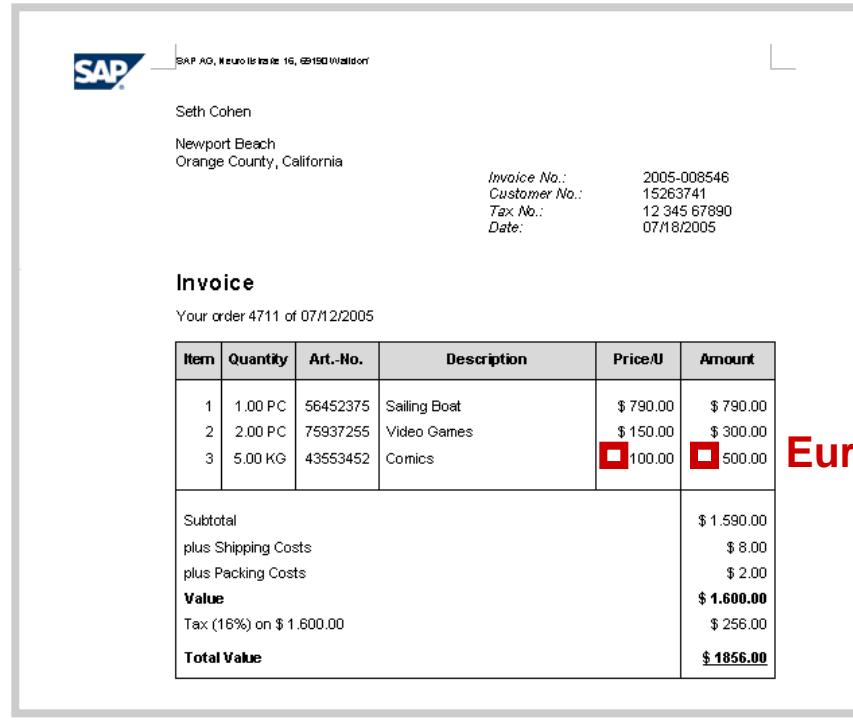
The EXECUTE method contains the main validation logic.

- **IO\_READ:** Reference to an access object to access instance data
- **EO\_MESSAGE:** Message object, for returning messages related to the validation
- **ET\_FAILED\_KEY:** Keys from the `IT_KEY` that do not match the validation criteria. If we implement an action validation, the corresponding action is not executed on items contained in `ET_FAILED_KEYS` (e.g. if the item quantity of one instance is negative).



# Consistency Validations

# Consistency Validations



A screenshot of an SAP invoice document. The header includes the SAP logo, address (SAP AG, Neuro Islandstr. 16, 69190 Walldorf), and recipient information (Seth Cohen, Newport Beach, Orange County, California). The invoice details show an invoice number (2005-008546), customer number (15263741), tax number (12 345 67890), and date (07/18/2005). The main body is titled 'Invoice' and shows an order from 07/12/2005. It lists three items: Sailing Boat (\$790.00), Video Games (\$150.00), and Comics (\$100.00). The subtotal is \$1,590.00, plus shipping costs (\$8.00), plus packing costs (\$2.00), totaling \$1,600.00. Tax (16%) on \$1,600.00 is \$256.00, resulting in a total value of \$1,856.00. A red question mark 'Euro, dollar, ... ?' is overlaid on the right side of the invoice.

Item	Quantity	Art.-No.	Description	Price/U	Amount
1	1.00 PC	56452375	Sailing Boat	\$ 790.00	\$ 790.00
2	2.00 PC	75937255	Video Games	\$ 150.00	\$ 300.00
3	5.00 KG	43553452	Comics	<input type="checkbox"/> 100.00	<input type="checkbox"/> 500.00

Subtotal  
plus Shipping Costs  
plus Packing Costs  
**Value**  
Tax (16%) on \$ 1,600.00  
**Total Value**

Euro, dollar, ... ?

## Motivation for consistency validation

- Consistency validations check whether a node instance is consistent with respect to the consistency criteria imposed by the business requirements.
- **Example:** If the node instance `ITEM` has the attribute `PRICE_AMOUNT` specified, but the attribute `PRICE_CURR_CODE` is initial, the node instance is inconsistent.

# Consistency Validations

---

Consistency validations check the consistency of node instances.

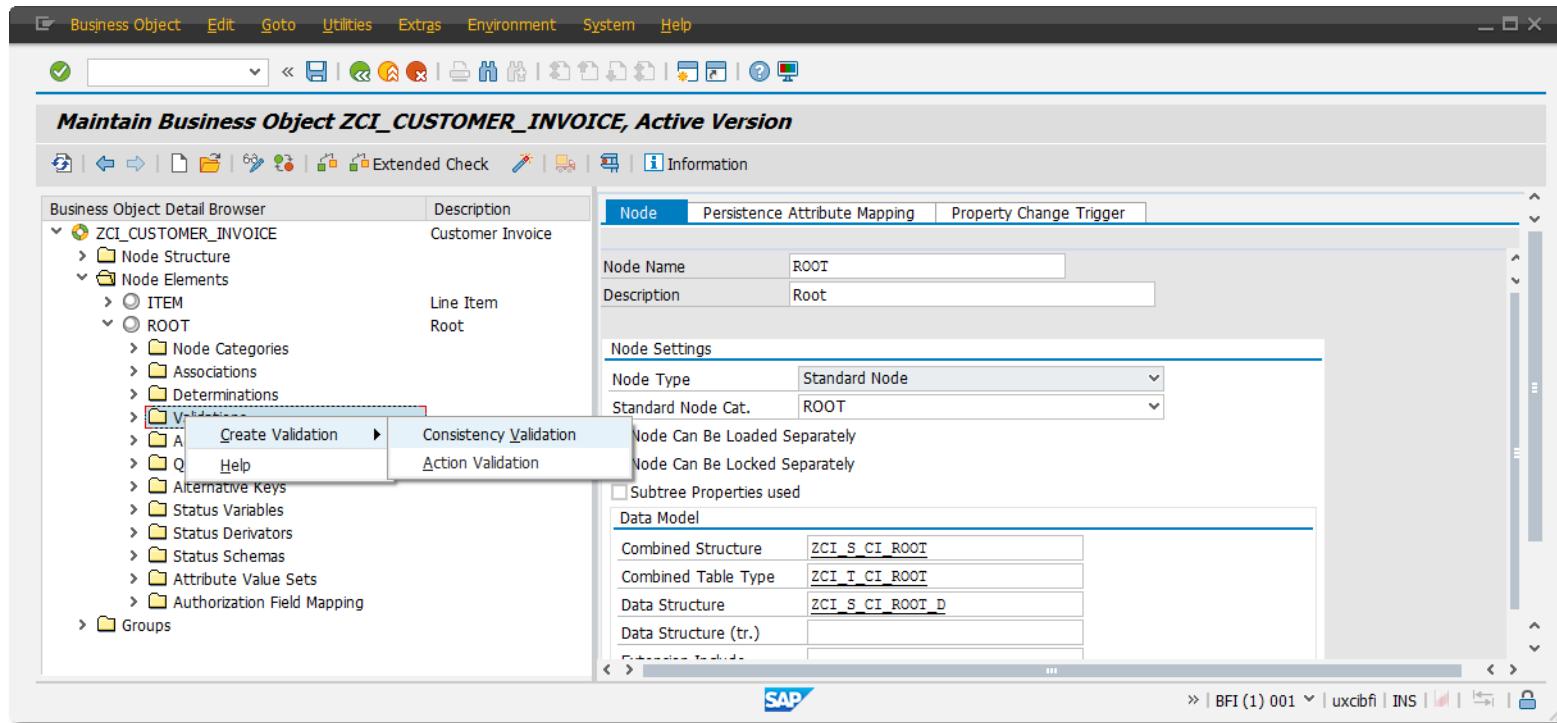
Consistency validations only return error messages and the keys of the failed instances (i.e. instances that do not match the consistency criteria).

Consistency validations can be configured to be executed at the following times:

- If a consumer executes a check and the request node of the consistency validation is in the check scope (e.g. check or check & determine).
- If there a change is made to the request node of the consistency validation (e.g. create, update, delete).
- If a consistency group that contains this consistency validation is executed.

# Consistency Validations

## How to Create Them



The guided procedure starts automatically after a consistency validation is created, or can be executed afterwards from the context menu.

# Consistency Validations

## Configuration

Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version

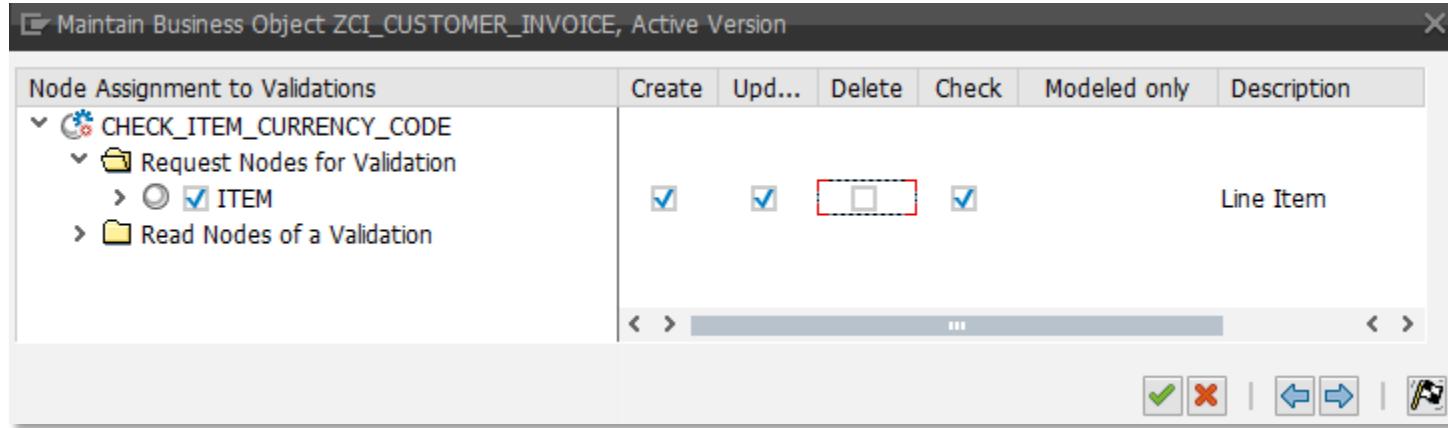
Validation Name	CHECK_ITEM_CURRENCY_CODE
Description	
Validation Settings	
Node	ITEM
Validation Cat.	Consistency Check
Class/Interface	ZCL_CI_V_CHECK_CURRENCY_CODE
<input checked="" type="checkbox"/> Check method implemented	
<input checked="" type="checkbox"/> Check Delta method implemented	
Test Data	
Test Data Container	
Variant	
Administrative Data	
Author	
Creation Date	
Last Changed By	
Changed On	

Buttons at the bottom: | |

Define a name, description, and implementing class (the implementing class can be created from this screen using forward navigation).

# Consistency Validations

## Request Nodes

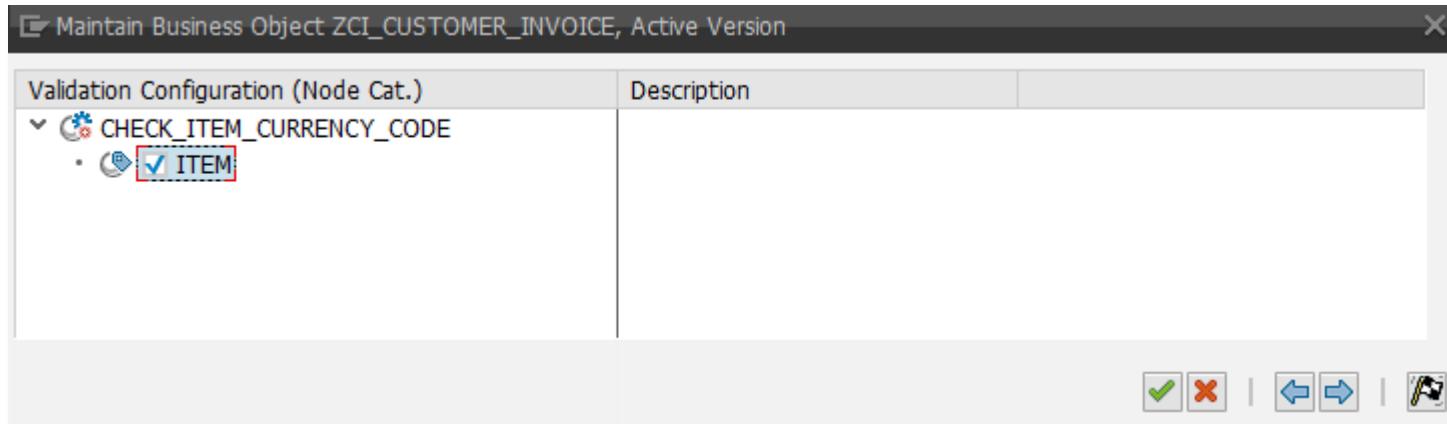


### Define the request node and trigger condition:

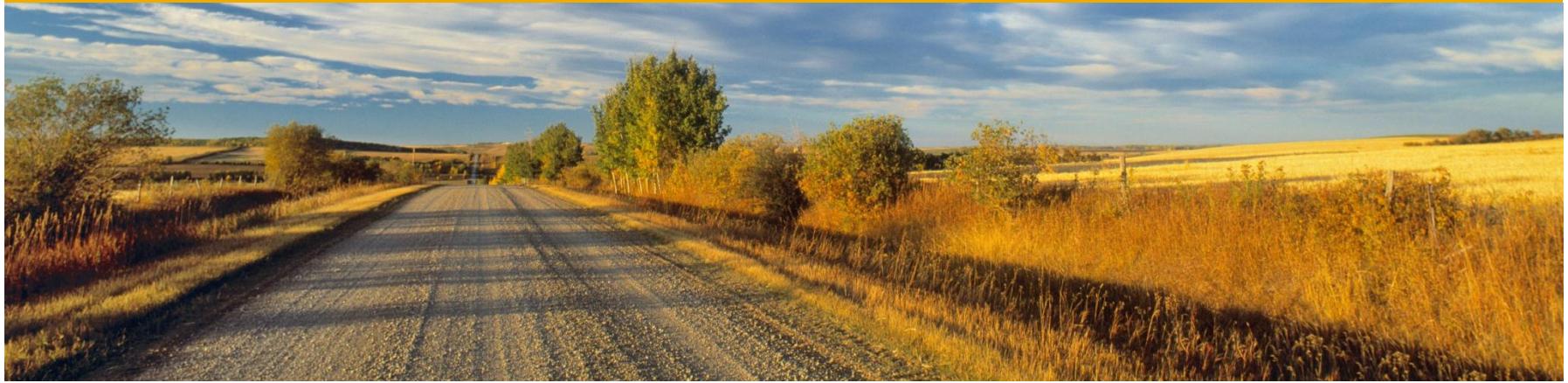
- **Request Node:** Node that triggers the validation execution if the trigger condition is fulfilled.
- **Request Condition:** Defines which kind of operation of a request node's instance triggers the validation execution. This can be triggered by a change of data (create, update, delete) or by the user executing a check on the request node.

# Consistency Validations

## Node Category Assignment



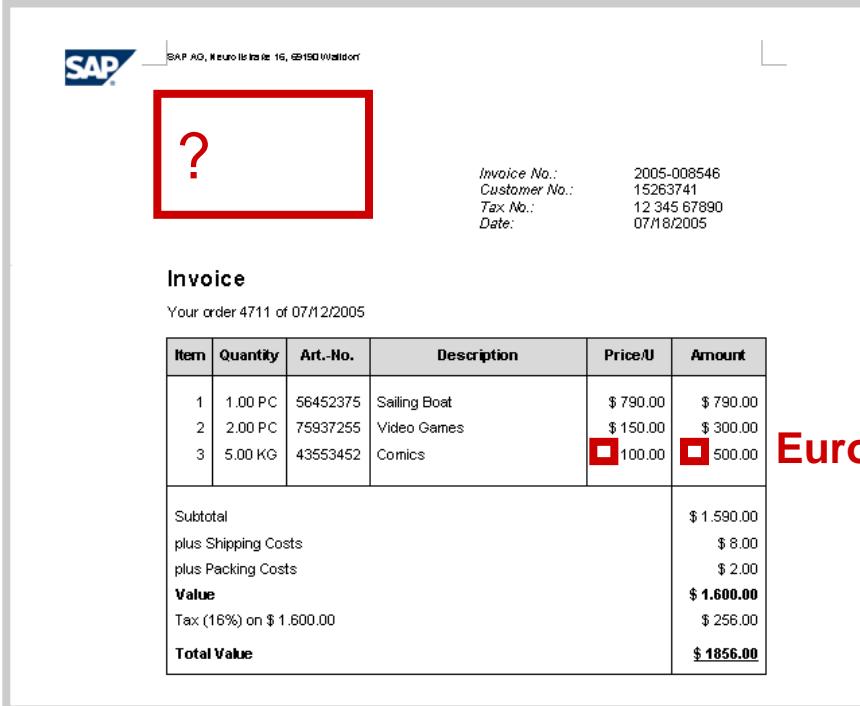
Activate the validation for an existing node category.



# Consistency Groups

# Consistency Groups

## Motivation



The screenshot shows an SAP invoice document. At the top left is the SAP logo and the text "SAP AG, Neurolektarstrasse 16, 69150 Walldorf". A large red rectangular box highlights the currency code field, which contains a question mark "?". To the right of this box, the invoice details are listed:

Invoice No.:	2005-008546
Customer No.:	15263741
Tax No.:	12 345 67890
Date:	07/18/2005

Below this is the heading "Invoice" followed by "Your order 4711 of 07/12/2005". The main body of the invoice contains a table of items:

Item	Quantity	Art.-No.	Description	Price/U	Amount
1	1.00 PC	56452375	Sailing Boat	\$ 790.00	\$ 790.00
2	2.00 PC	75937255	Video Games	\$ 150.00	\$ 300.00
3	5.00 KG	43553452	Comics	\$ 100.00	\$ 500.00

At the bottom of the item table, there are additional line items for "Subtotal", "plus Shipping Costs", "plus Packing Costs", and "Value". The "Value" section includes "Tax (16%) on \$ 1.600.00" and "Total Value". The final total value is shown as "\$ 1856.00". To the right of the invoice, the text "Euro, dollar, ... ?" is written in red.

## Motivation for Consistency Groups

- Only the instances of the invoice business object that are in a consistent state are to be saved.
- **Example:** If there is no currency code, the invoice cannot be saved.

# Consistency Groups

## Motivation

---

### Consistency groups contains a set of:

- **Consistency validations:** Only their failed keys are evaluated; messages are not relevant.
- **Delegated nodes:** Their error messages are evaluated (more about delegated nodes is described in *BS-BOPF Extended Training*).

### General properties of consistency groups

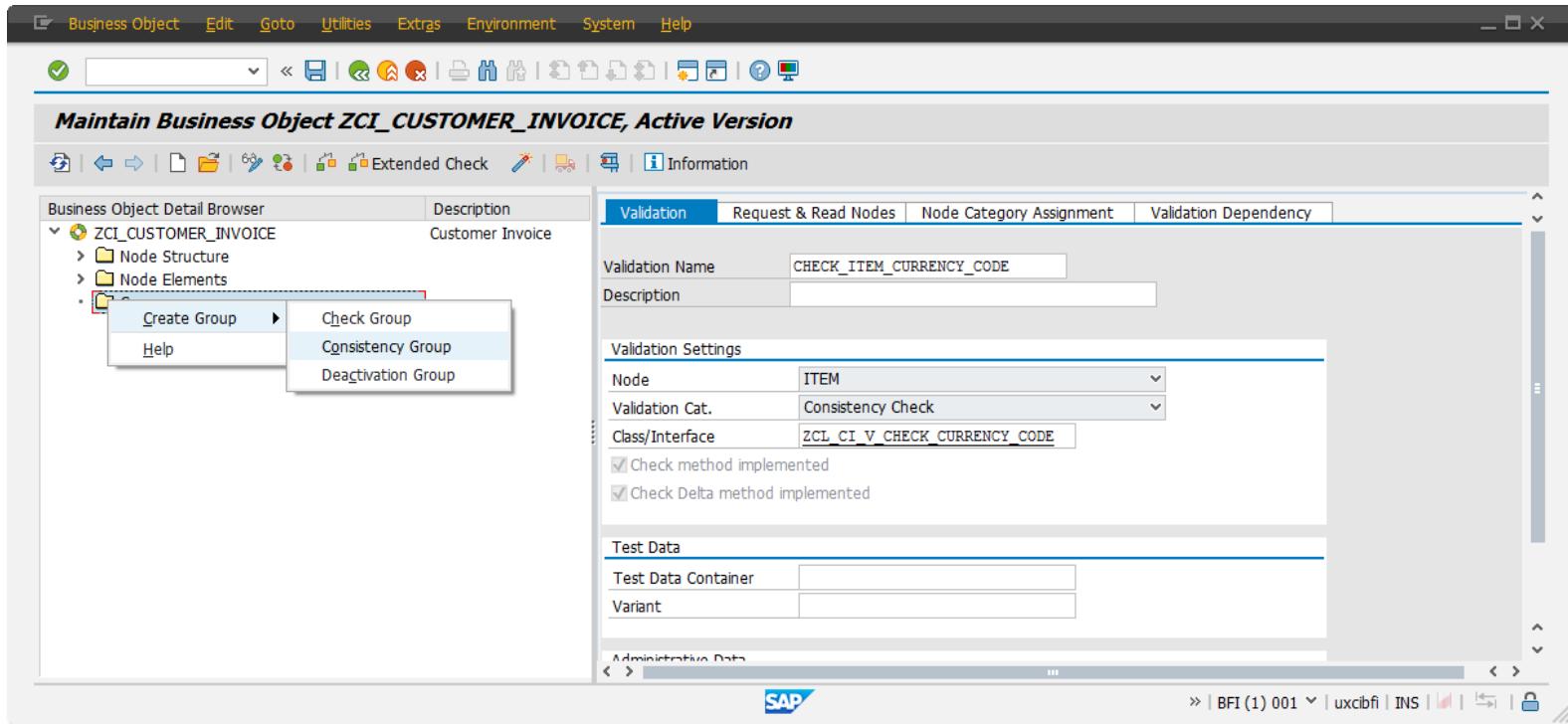
- The contained consistency validations are still executed if their trigger condition is met (regardless of which consistency group they belong to).
- If a consistency group is executed, each group “member” is executed.

### Consistency groups are either used to...

- **Prevent inconsistent BO instances from being saved** (described in Basic Training here)  
To reuse consistency validations (to prevent inconsistent instances from being saved), consistency groups can be used.  
If at least one failed key is returned from a validation of a group, no saves are possible.
- **Or to set a consistency status** (described in BS-BOPF Extended Training)  
Saves are not prevented, but a status variable is set instead.

# Consistency Groups

## How to Create Them



Consistency groups are created using a context menu entry.

Afterwards the guided procedure for configuration starts automatically.

# Consistency Groups

## Configuration

Maintain Business Object ZCI\_CUSTOMER\_INVOICE, Active Version

Group	ITEM_CONSISTENCY
Description	
Group Settings	
Group Category	Consistency Group
Node	Nothing chosen
Action	Nothing chosen
Status Variable	Nothing chosen
Administrative Data	
Author	
Creation Date	
Last Changed By	
Changed On	

Buttons at the bottom: Save (green checkmark), Cancel (red X), Back (left arrow), Forward (right arrow), and Help (magnifying glass).

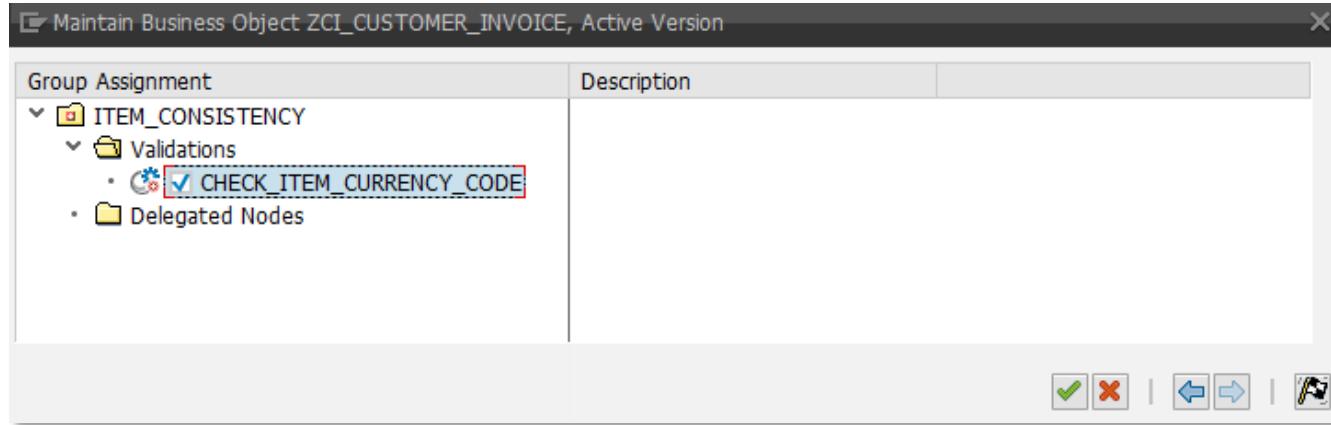
Keep these fields empty to define a consistency group that prevents saves.

- Start the guided procedure.
- Enter a name and description for the group.

**Important:** Since the consistency group must prevent saves and not set a consistency status, the fields “Node”, “Action”, “Status Attr. Name”, and “Status Variable” must stay blank.

# Consistency Groups

## Configuration



Assign the consistency validations that are to be part of this group.

# Overview

## Examples of Validation Configurations

	<b>Execution of Check Core Service</b>	<b>Before a certain Action is executed</b> (action is not executed for instances failing the validation)	<b>Before a Modification is executed by the framework</b> (modifications are rejected in case of failing the validation)	<b>Immediately after a Modification has been executed by the framework</b> (no modification rejection, only messages can be returned)	<b>At Save</b> (save rejected in case of failing the validation)
<b>Action Validation</b> (configured to an own action)		X			
<b>Action Validation</b> (configured to Create/Update/Delete)			X		
<b>Action Validation*</b> (configured to Save)					X
<b>Action Validation</b> (configured to Create/Update/ Delete and Save)			X		X
<b>Consistency Validation**</b> (not member of a group)				X	
<b>Consistency Validation***</b> (which is member of a Save Preventing Consistency Group)				X	X
<b>Consistency Validation</b> (configured to Check and Save)	X				X

\*) Ensures consistent persisted instance data

\*\*) Usability advantage as user can always save and leave the application (e.g. go for lunch), but allows to persist inconsistent data

\*\*\*) Validations are executed twice which might cause performance issues for very expensive validations.



# Thank you

# © 2012 SAP AG. All rights reserved.

---

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, PowerPoint, Silverlight, and Visual Studio are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, z10, z/VM, z/OS, OS/390, zEnterprise, PowerVM, Power Architecture, Power Systems, POWER7, POWER6+, POWER6, POWER, PowerHA, pureScale, PowerPC, BladeCenter, System Storage, Storwize, XIV, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, AIX, Intelligent Miner, WebSphere, Tivoli, Informix, and Smarter Planet are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the United States and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are trademarks or registered trademarks of Adobe Systems Incorporated in the United States and other countries.

Oracle and Java are registered trademarks of Oracle and its affiliates.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems Inc.

HTML, XML, XHTML, and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Apple, App Store, iBooks, iPad, iPhone, iPhoto, iPod, iTunes, Multi-Touch, Objective-C, Retina, Safari, Siri, and Xcode are trademarks or registered trademarks of Apple Inc.

iOS is a registered trademark of Cisco Systems Inc.

RIM, BlackBerry, BBM, BlackBerry Curve, BlackBerry Bold, BlackBerry Pearl, BlackBerry Torch, BlackBerry Storm, BlackBerry Storm2, BlackBerry PlayBook, and BlackBerry App World are trademarks or registered trademarks of Research in Motion Limited.

Google App Engine, Google Apps, Google Checkout, Google Data API, Google Maps, Google Mobile Ads, Google Mobile Updater, Google Mobile, Google Store, Google Sync, Google Updater, Google Voice, Google Mail, Gmail, YouTube, Dalvik and Android are trademarks or registered trademarks of Google Inc.

INTERMEC is a registered trademark of Intermec Technologies Corporation.

Wi-Fi is a registered trademark of Wi-Fi Alliance.

Bluetooth is a registered trademark of Bluetooth SIG Inc.

Motorola is a registered trademark of Motorola Trademark Holdings LLC.

Computop is a registered trademark of Computop Wirtschaftsinformatik GmbH.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, SAP HANA, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase Inc. Sybase is an SAP company.

Crossgate, m@gic EDDY, B2B 360°, and B2B 360° Services are registered trademarks of Crossgate AG in Germany and other countries. Crossgate is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.