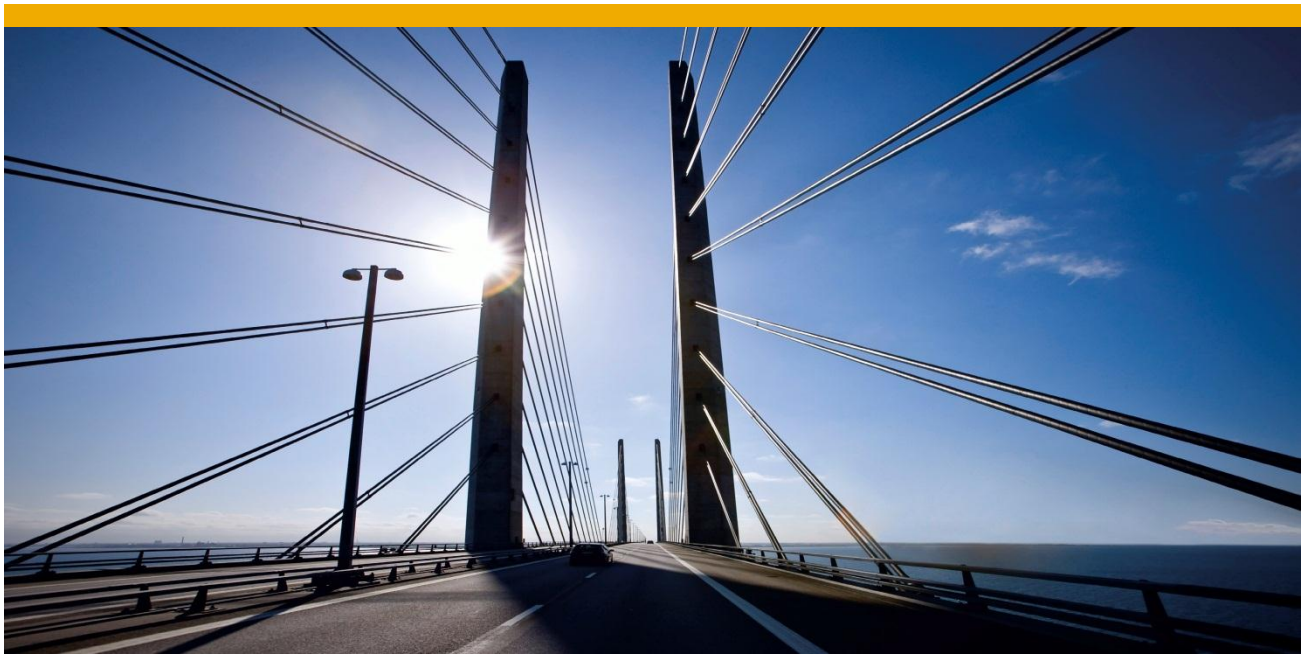

How to use authority checks in Business Object Processing Framework

Summary

This tutorial aims to provide you with the knowledge required for embedding authorization checks in a Business Object. The resulting Business Object will deny the access to data and the execution of actions the user is not allowed to.

Level of complexity:	**Intermediate**
Time required for completion:	**30 min**

Author:	**Thea Hillenbrand**
Company:	**SAP AG**
Created on:	**10 September 2013**

TABLE OF CONTENTS

BEFORE YOU START

Objectives

By the end of this Getting Started, you will
- be able to define authorization objects to be used in BOPF
- be able to assign authorization objects to a node of a business object
- understand how the authorizations in BOPF, the user roles and the runtime work together

The objects we use and create in this tutorial are simplified versions of the sample BO /BOPF/EPM_SALES_QUOTE. We continue on the example used in the Getting Started with BOPF tutorial.

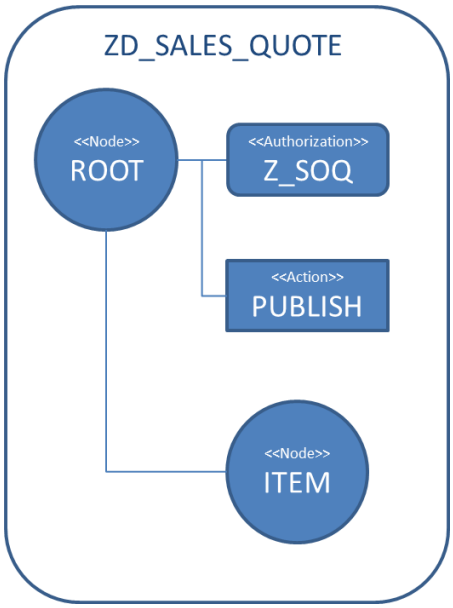

Fig. 1: The BO Structure used in this tutorial

Prerequisites

In order to be able to perform this tutorial, the following prerequisites have to be fulfilled.

Systems, releases, and authorizations

- BOPF is part of the Business Suite Foundation Layer. The feature is available in
  - SAP Business Suite EHP7 SP03.
- For modifying a Business Object, your SAP user requires the developer authorization profile (S_DEVELOP authorization object).
- For defining authorization objects, your SAP user requires the authorization for the transactions SU20 and SU21.
- In most cases it is unlikely that you have the permissions to create user roles with transaction PFCG and assign them to users. If you can create the roles described in the last chapter you can test the permissions in the BOPF test tool (BOBT) or in your own UIs.

Knowledge

- You understand the SAP authorization and role concept
- Knowledge of the transactions SU20, SU21, PFCG
- Knowledge of BOPF
- Having worked through the Getting Started with Business Object Processing Framework

DEFINE AUTHORIZATION OBJECTS

With the SAP authorization and role concept it is possible to check if a user is allowed to perform a certain activity in general or to check if a user is allowed to manipulate certain data. We call the first check **static** because it is independent from concrete data values. The second check is called **instance-based** as the result of the check depends on the concrete node data: a user may be allowed to work on the sales quotes of his business partner but not on the business partners of his colleagues.

Both types of checks are supported by BOPF by the same authorization object. It just has to contain the predefined authorization fields ***ACTVT*** and ***BO_SERVICE.*** These fields are used for static and instance-based checks.
In the case of instance based checks you need to define the fields for the values to be checked. In our example this is the business partner id.

Prerequisites

You know the transaction SU20 to define authorization fields and SU21 to define authorization objects.

Procedure

In this step you will create the authorization field ***Z_SQ_BP_ID*** for the business partner id and the authorization object ***Z_SOQ*** to be used in the business object ***ZD_SALES_QUOTE***.

Launch transaction SU20

The transaction SU20 provides the design time for authorization fields. Select the **Create** button and enter the authorization field name ***Z_SQ_BP_ID.*** Use the data element ***SNWD_PARTNER_ID*** as reference field.

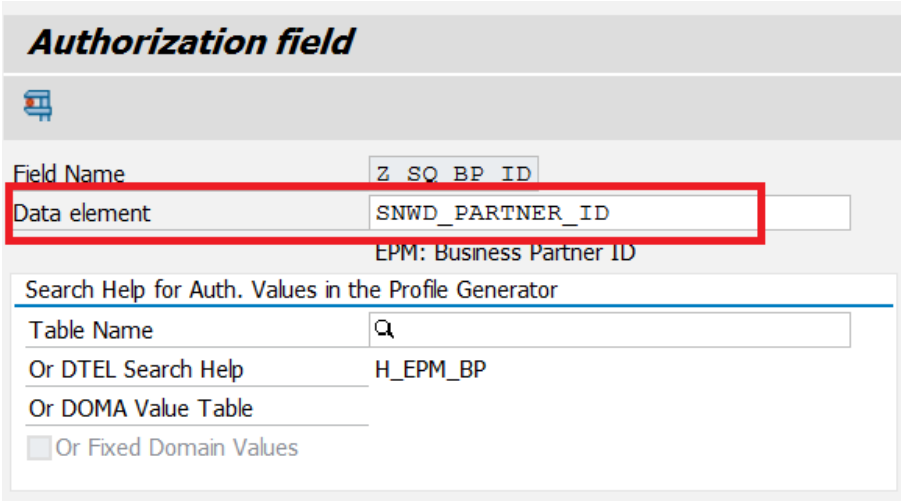

Fig. 2: Authorization field for the business partner id

Save the definition and assign it to your local objects.

Launch transaction SU21

The transaction SU21 provides the design time for authorization objects. Select in the **Create** menu button the entry *authorization object.* Confirm the information popup and continue on the next screen. Enter the object name ***Z_SOQ*** and a description. Authorization objects are classified. Enter a classification for test purposes in the field *Class* as you are defining a test object. Assign the authorization fields ***ACTVT***, ***BO_SERVICE*** and ***Z_SQ_BP_ID.***

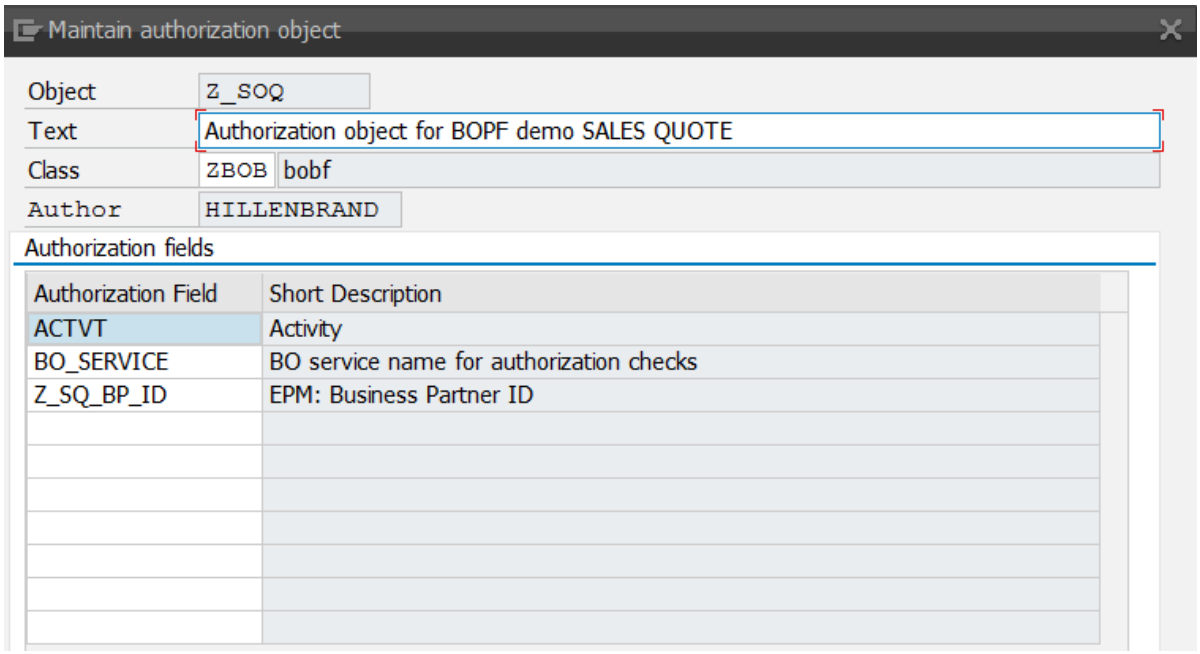

Fig. 3: Authorization object

Save the definition.

Result

You have created the authorization object which can be used in the business object ***ZD_SALES_QUOTE***.

ASSIGN THE AUTHORIZATION OBJECT TO A BUSINESS OBJECT

Authorization objects are assigned to a node – normally to the root node, but this depends on your business requirements. It is possible to assign authorization objects to each node.
If you assign an authorization object to a node, it applies to the complete subtree unless there is a different assignment to a child node.

Prerequisites

You have defined the business object ***ZD_SALES_QUOTE*** as described in the Getting Started with Business Object Processing Framework.

Procedure

In this step you will assign the authorization object to the root node of the business object ZD_SALES_QUOTE and map the fields. Embedded authorizations are an advanced feature which you cannot define in the transaction BOB, but you have to use the transaction BOBX (BO Builder eXpert mode)[1].

Enable authorization checks

Launch transaction BOBX and select the business object ***ZD_SALES_QUOTE*** via double click. Open the business object in change mode and mark the check box *Business Object has Authorization checks*. This flag enables the authorization checks.
Save the definition.

[1] The transaction BO Builder eXpert is available in SAP Business Suite Foundation 7.47 SP03.

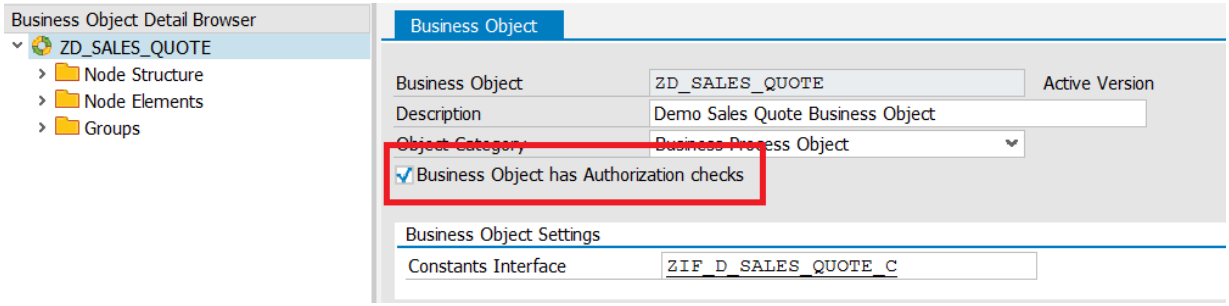

Fig. 4: Enablement for authorization checks

Assign authorization object to root node

Select the root node and mark the check box *Node has own checks*. The system proposes the library class /BOBF/CL_LIB_AUTHORITY_CHECK, accept it. Save the definition.

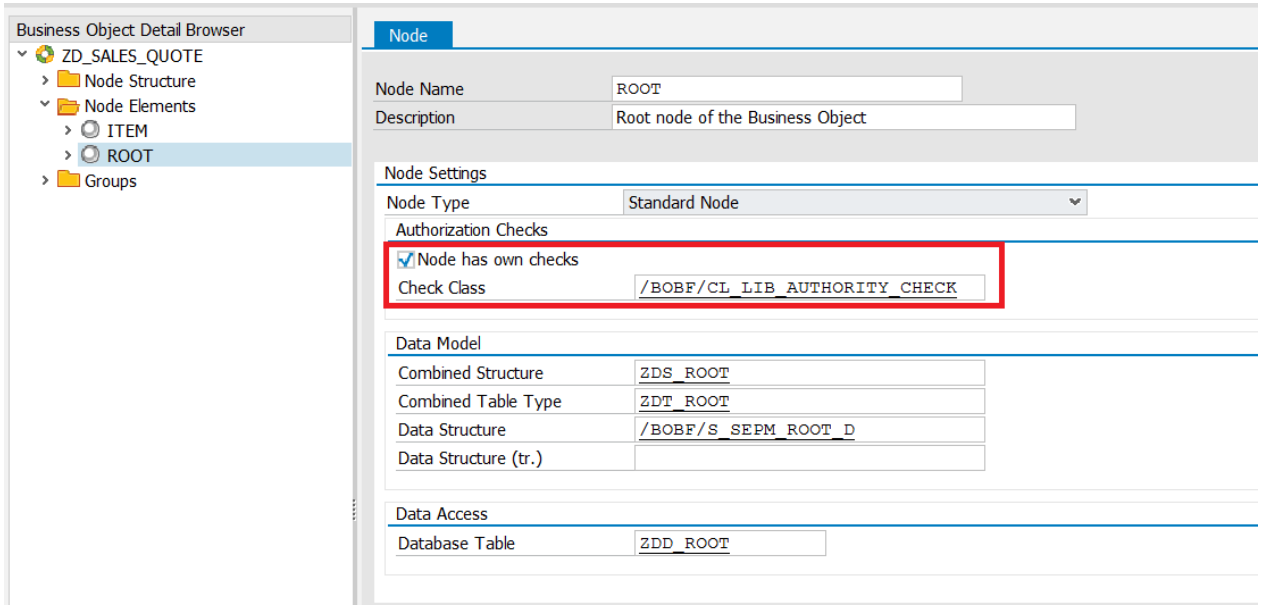

Fig. 5: Select the root node for authorization checks

Select the folder *Authorization Objects* in the navigation pane and choose the function *Assign Authorization Object* in the context menu. Enter ***Z_SOQ*** as *Authorization Object for BO* and save the definition.

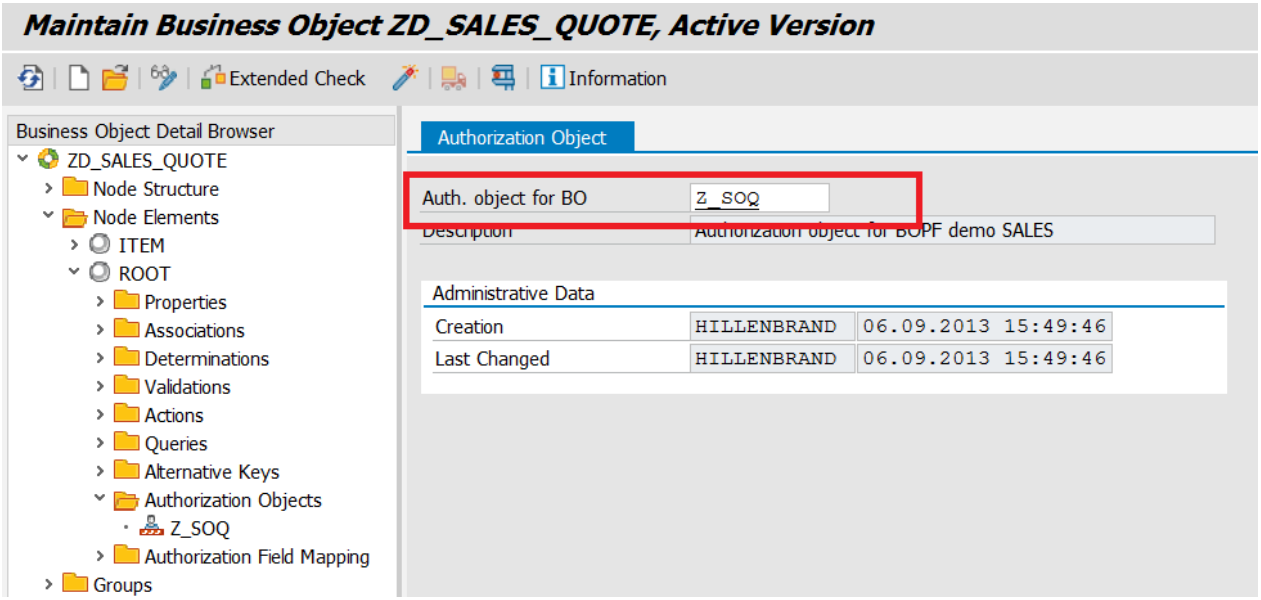

Fig. 6: Authorization Object Assignment

Define the field mapping

You can now proceed with the field mapping. Here you relate the fields of the authorization object with the fields of the BO. This is necessary to call the authorization check with the correct node data at runtime. The fields ACTVT and BO_SERVICE are automatically mapped. The framework uses this assignment for static checks. You have only to map the fields defined by you for instance based checks.
Select *Authorization Field Mapping* in the navigation pane and choose *Create Authorization Field Mapping* in the context menu.
Select the authorization object ***Z_SOQ*** in the drop down list box for *Authorization Object for BO* and the field ***Z_SQ_BP_ID*** in the drop down list box for *Authorization Field for BO*. Select the Business Partner ID BP_ID, an element of the persistent structure of the root node, as target attribute.Save the definition.

> *Hint: you can also follow an association. This is useful to define authorization checks with attributes of another node – for example to check that users are only allowed to display the sales quotes of the business partners in a certain region. The region is an attribute of the business partner, not of the root node of the sales quote.*

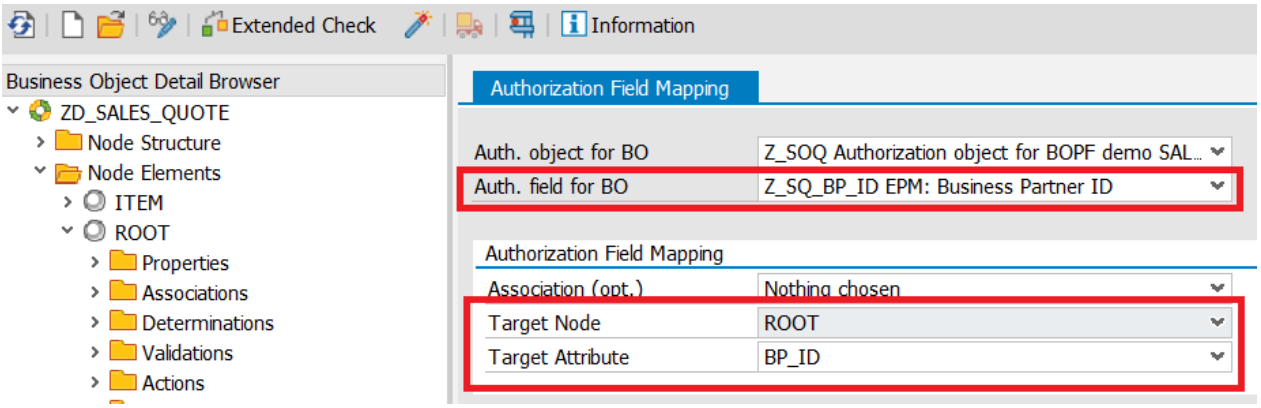

Fig. 7: Field mapping

Result

You have assigned the authorization object to the business object. Authorization checks are now executed by the framework. How the BO services and actions relate to the user profile is described in the next chapter.

RUNTIME

Behavior according to the user profile

In the previous steps we assigned an authorization object to a BO. The runtime behavior depends on the user profile. In the following we discuss three different user profile settings.

No permissions

If no role or profile is assigned to the user, the user has no authorizations for the BO. So any activity is stopped with an error message, for example 'No authority to display sales quote <sales_quote>' or 'No authority to publish sales quotes'.

Role for display

With the following user profile the user can display (Activity 03) the sales quotes related to the business partner ids starting with 1000. He cannot modify them or execute any action.

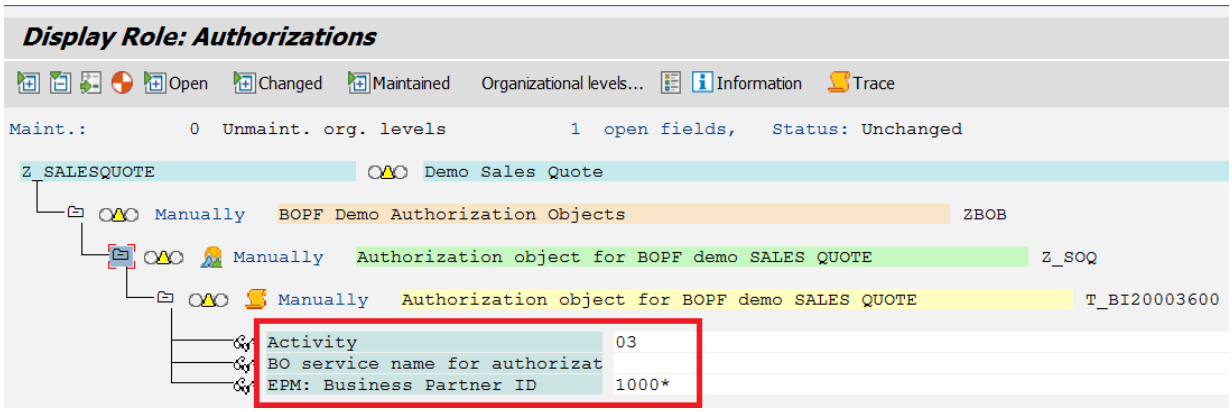

Fig. 8: Role to display Sales Quotes for certain business partners

The following role definition provides access to all sales quotes in display mode.

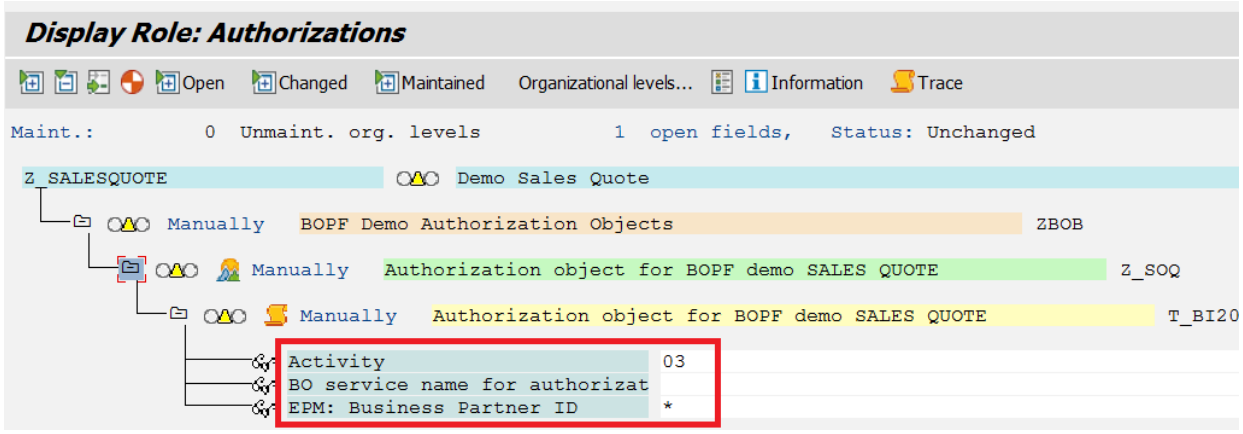

Fig. 9; Role to display Sales Quotes for all business partners

Role to publish a sales quote

To grant the permission to execute a BO action use the activity execute (16) and assign the action name to the BO_SERVICE field.

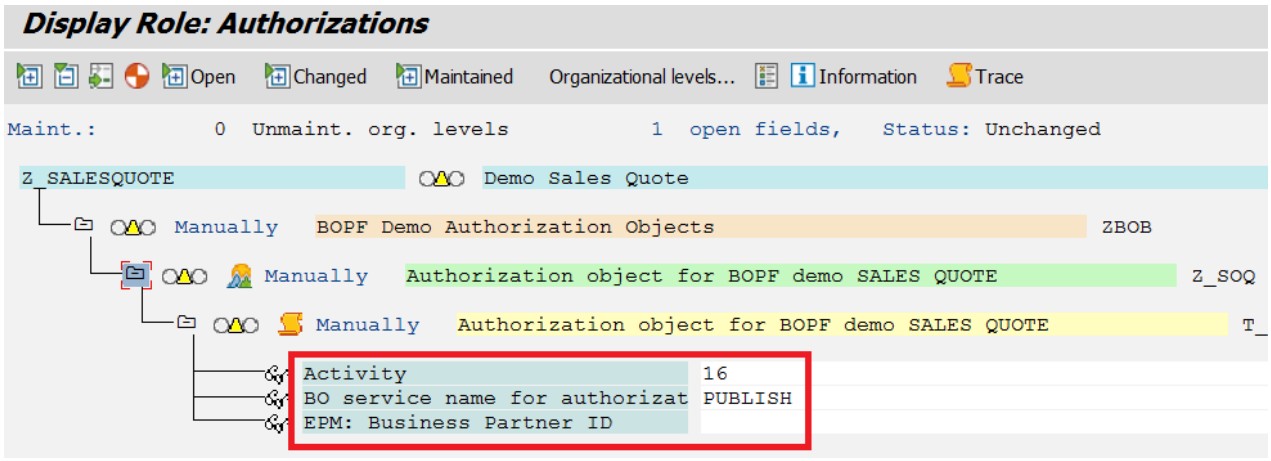

Fig. 10: Role to publish all Sales Quotes

Relation between BO service methods and the fields ACTVT and BO_SERVICE

The following table provides an overview on how the BO service methods relate to the authorization fields ACTVT and BO_SERVICE. For example, if the user triggers the MODIFY method by his user input, the framework checks the permissions for the activities create, change and delete. If he wants to execute an action, the system checks the permissions for the fields ACTVT with the activity execute and BO_SERVICE with the action name.

BO method	ACTVT	BO_SERVICE
CHECK_ACTION	DISPLAY (03)	
CHECK_AND_DETERMINE	CHANGE (02)	
CHECK_CONSISTENCY	CHECK (39)	
CONVERT_ALTERN_KEY	DISPLAY (03)	
DO_ACTION	EXECUTE (16)	<action name>
MODIFY	CREATE/CHANGE/DELETE (01/02/06)	
RETRIEVE	DISPLAY (02)	
RETRIEVE_BY_ASSOCIATION	DISPLAY (02)	
RETRIEVE_CODE_VALUE_SET	DISPLAY (02)	
RETRIEVE_DEFAULT_ACTION_PARAM	DISPLAY (02)	
RETRIEVE_DEFAULT_NODE_VALUES	DISPLAY (02)	
RETRIEVE_DEFAULT_QUERY_PARAM	DISPLAY (02)	
RETRIEVE_PROPERTY	DISPLAY (02)	
QUERY	QUERY	<query name>

Performance

In the BOPF runtime we implemented several performance optimization strategies: the canonical one reduces the calls in buffering the results. In addition we implemented an equivalence group design, meaning that we execute the check for instances which are identical with respect to the check relevant attributes only once. And last but not least the standard implementation uses the priviledged mode. This means, that authorizations are only checked once at the beginning of a request. The user is then authorized to access the data necessary to fulfill the request – even if data from different nodes or business objects is requested - without further checks.

Result

This tutorial walked you through defining an authorization object and assigning it to a BO. The pure doing is very simple and can be done at a late phase of the project. So you can carefully discuss data protection needs with your stakeholders using the user interfaces of the application. The framework cares for executing the checks consistently.

But you must understand the SAP authorization and role concept as well as the relation between BO methods and the fields ACTVT and BO_SERVICE to come up with a meaningful authorization concept for your application.

www.sap.com