



Getting Started with Business Object Processing Framework

Summary

This tutorial aims to provide you with the basic knowledge required for creating and testing a BOPF Business Object. The resulting Business Object will consist of two nodes and a simple piece of business logic, represented by an action.

Level of complexity:	Beginner
Time required for completion:	30 minutes

Author:
Company:
Created on:

Carsten Schminke
SAP AG
20 August 2013

TABLE OF CONTENTS

BEFORE YOU START	3
Objectives.....	3
Prerequisites	3
Systems, Releases, and Authorizations	3
Knowledge	3
CREATE A BUSINESS OBJECT	4
Procedure	4
Launch the Business Object Builder (BOB).....	4
Start the Business Object Creation Wizard	4
Define Business Object Name and Description	4
Define Constants Interface Name	5
Define the Root Node and its Attributes	6
Choose names for root node types	8
Finish the BO creation procedure	8
Result.....	9
ADD A SUBNODE	9
Prerequisites	9
Procedure	9
Start Node Creation Wizard.....	9
Create subnode	9
Result	10
ADD AND IMPLEMENT AN ACTION.....	10
Prerequisites	10
Procedure	10
Start Action Creation Wizard	10
Define Name and Description of the Action.....	11
Define the Implementing Class.....	11
Finishing the wizard	12
Implement the Action	12
Result	13
TEST THE BUSINESS OBJECT	13
Prerequisites	13
Procedure	13
Start Business Object Test Environment.....	13
Create, Save and Query Node Instances.....	14
Execute the Action.....	17
Result	17

BEFORE YOU START

Objectives

By the end of this Getting Started document you will be able to

- Create a business object, including the root node
- Create a subnode
- Create an action
- Implement a simple action based on the BOPF API
- Test the functions of the new business object



Fig. 1: Structure of the Business Object to be created

Prerequisites

Before you perform this tutorial, make sure the following prerequisites are fulfilled.

Systems, Releases, and Authorizations

- BOPF is part of the Business Suite Foundation Layer and, therefore, included in the following SAP Business Suite releases:
 - SAP Business Suite EHP5, SP11
 - SAP Business Suite EHP6, SP05
 - SAP Business Suite EHP7, all SP
- To create a Business Object, your SAP user requires the developer authorization profile (S_DEVELOP authorization object).

Knowledge

- Basic knowledge in ABAP OO
- Experience with DDIC tools.

CREATE A BUSINESS OBJECT

In this step, you will create a sample Business Object (BO) “*SALES_QUOTE*”. This Business Object follows the semantics of the sales quote based on the NetWeaver Enterprise Procurement Model (EPM).

Procedure

Launch the Business Object Builder (BOB)

The transaction BOB provides the design time for creation of custom business objects as well as business object enhancements.

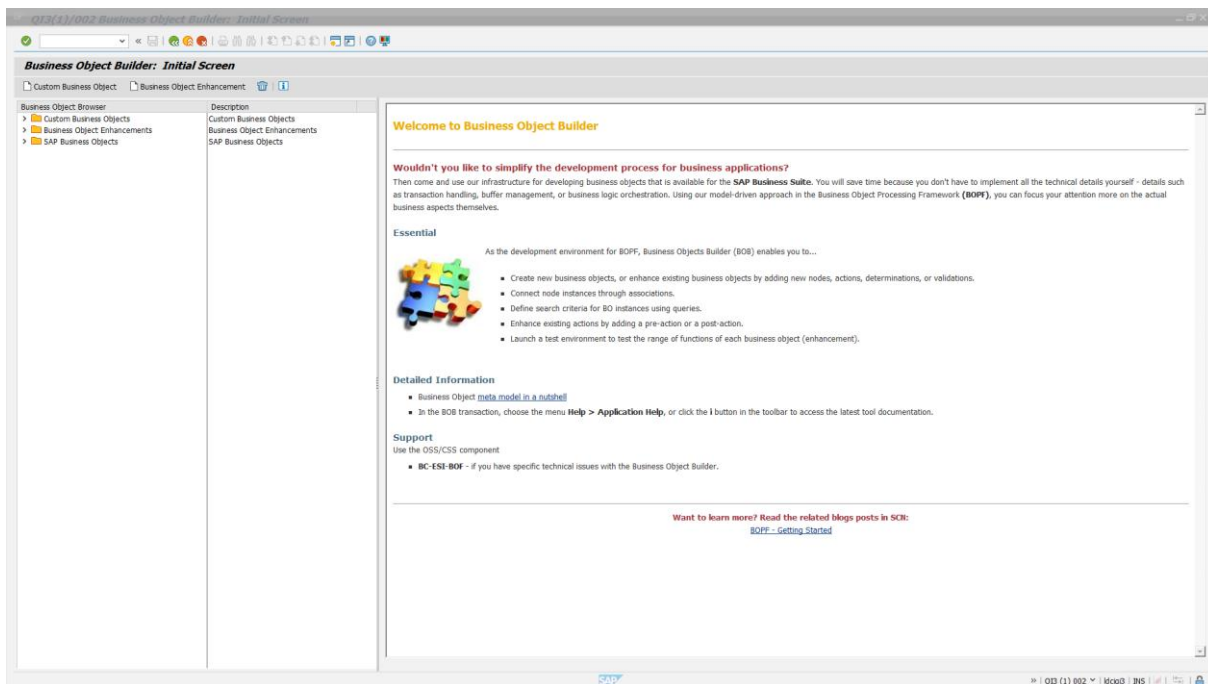


Fig. 2: Welcome page of the BOB transaction in a SAP customer system

On the left side of the initial screen, you see three categories of BOs that are currently available in your system:

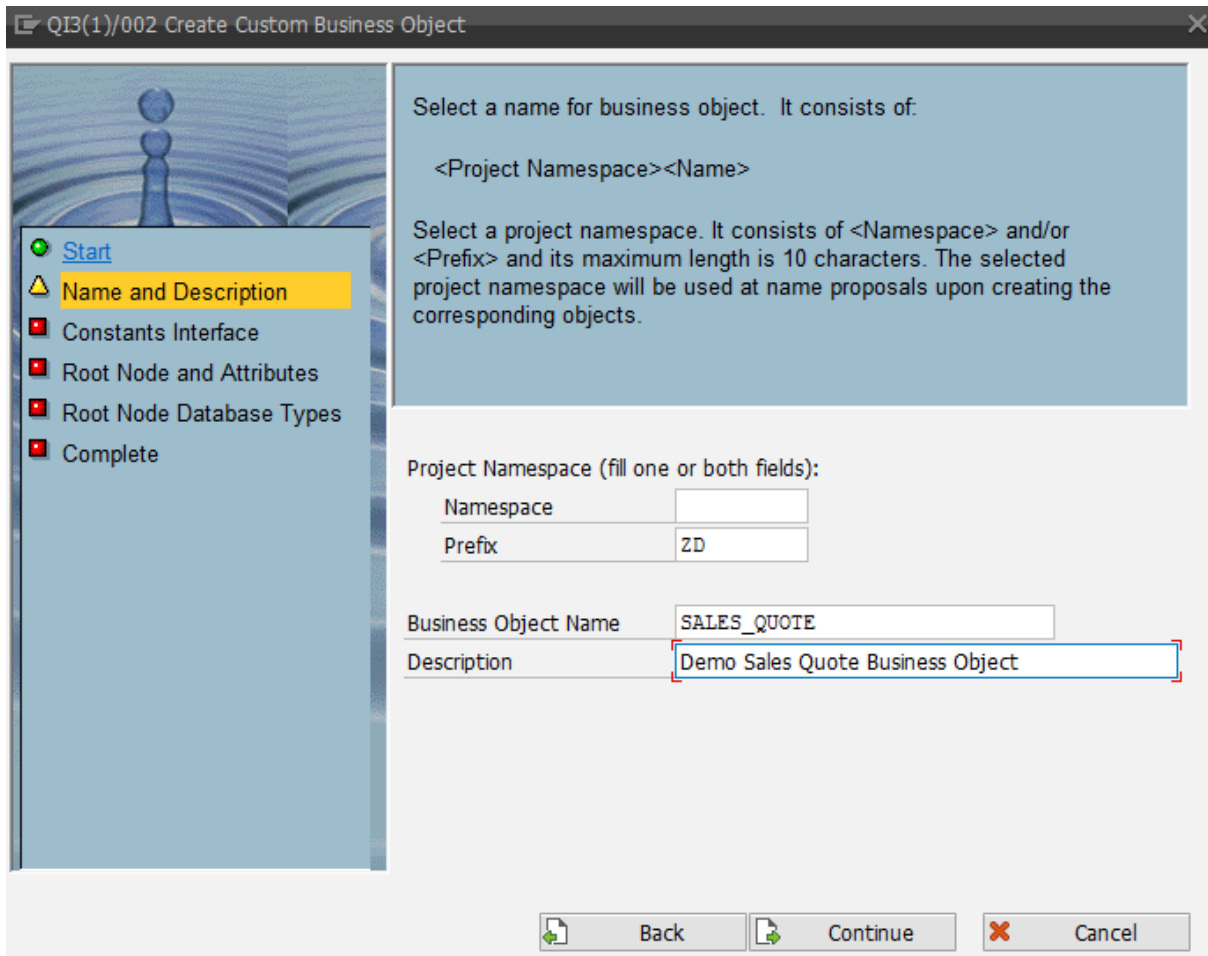
- Custom Business Objects: BOs created by the customer in their system
- SAP Business Objects: Extensible BOs that are delivered by SAP
- Business Object Enhancement: BOs that are enhancements to an existing business object

Start the Business Object Creation Wizard

Select **Create Custom Business Object** from the application toolbar. A wizard is started that will guide you through the necessary steps. Choose **Continue** to go to the first step.

Define Business Object Name and Description

In the first step you have to provide a project namespace, represented by a registered namespace and/or a prefix. The project namespace will be used as a prefix not only for the BO name but also for all other artifacts that belong to a BO.



QI3(1)/002 Create Custom Business Object

Select a name for business object. It consists of:

<Project Namespace><Name>

Select a project namespace. It consists of <Namespace> and/or <Prefix> and its maximum length is 10 characters. The selected project namespace will be used at name proposals upon creating the corresponding objects.

Project Namespace (fill one or both fields):

Namespace

Prefix

Business Object Name

Description

Back Continue Cancel

Fig. 3. Editing the name and the description for the BO

In the input fields *Business Object Name* and *Description* you provide the name describing the semantics of the new BO and a short text, respectively. Go to the next wizard step by choosing **Continue**.

Define Constants Interface Name

In this step, the technical name of the business object is shown and you have the possibility to define the name for the so-called Constants Interface.

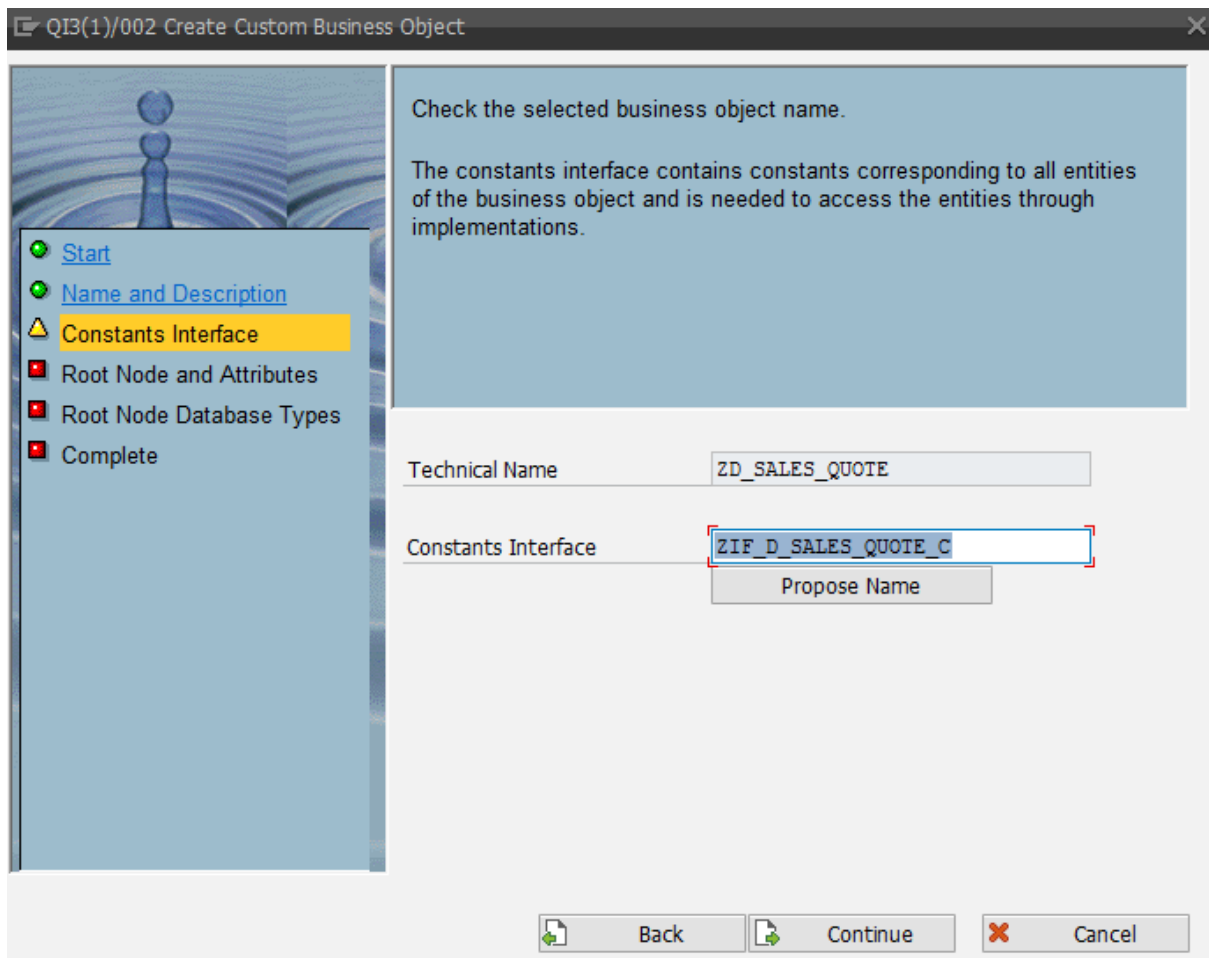


Fig. 4: Constants Interface is required for consuming services of the BO

The technical name of the BO is unique in the system and is automatically derived from the project namespace and the semantic name from the previous step.

The *Constants Interface* will be created automatically at the end of the wizard, and it will contain constants for identifying all the entities that are assigned to the BO. The constants interface plays a major role when consuming BO services. The tool already proposes a meaningful name; therefore nothing needs to be changed.

Continue with the next step.

Define the Root Node and its Attributes

The data model of a BO is represented as a hierarchical tree of nodes. The business object has exactly one root node that, in turn, can have multiple subnodes. Each subnode again can have subnodes, and so on.

In this step you specify the root node of the BO. The proposed name of the root node is ROOT, but this can also be changed individually. In addition, a short text can be provided.

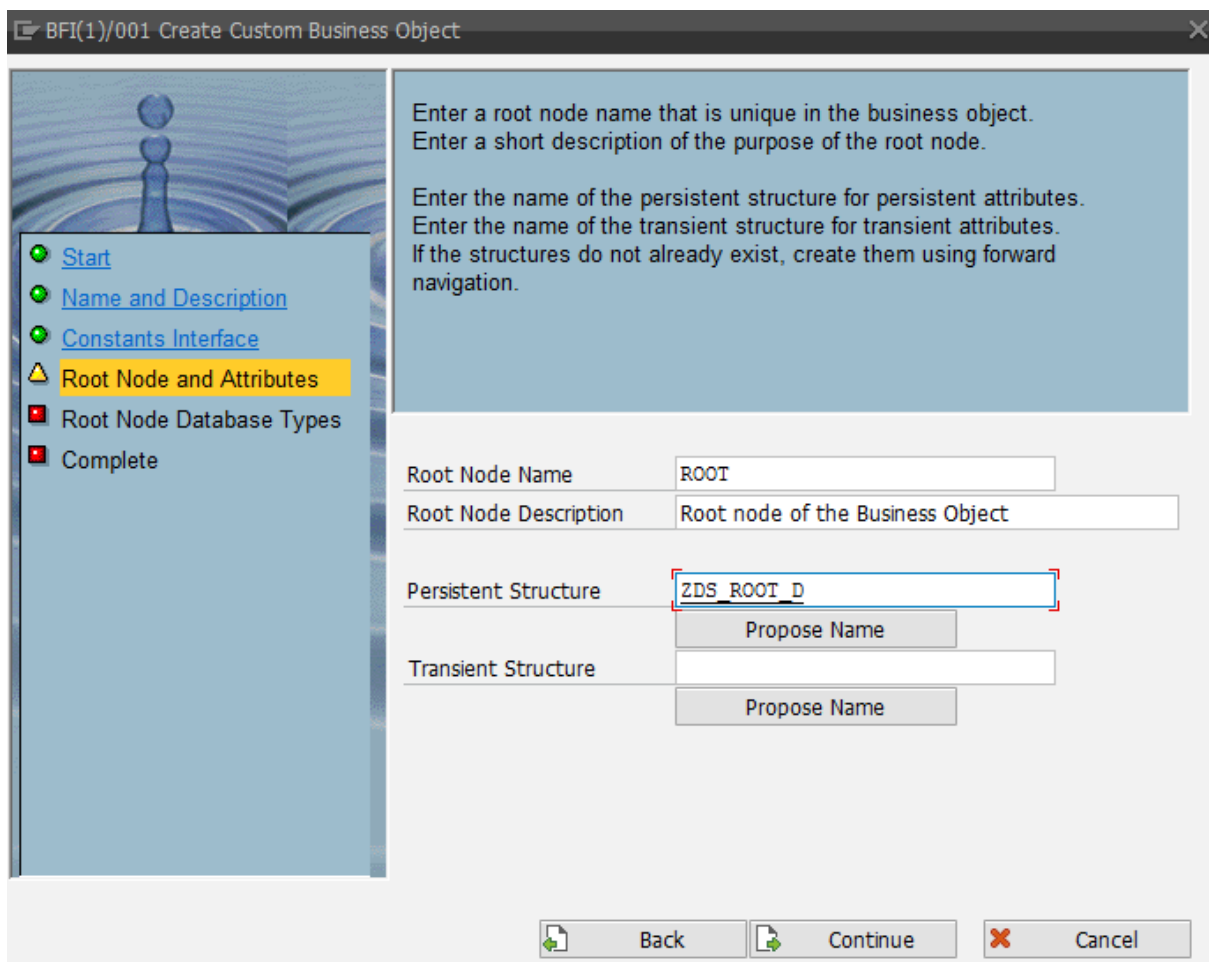


Fig. 5: Specifying the root node of the business object

A node consists of several attributes that are represented at the technical level by a Persistent and a Transient Structure. The *Persistent Structure* defines the attributes that are stored in the database whereas the attributes of the *Transient Structure* are calculated at runtime. Both structures are implemented as regular DDIC structures and are created in the DDIC transaction SE11. For our demo, we want to work with persistent node attributes only. Therefore we select **Propose Name** to get a meaningful name for the *Persistent Structure*. By double-clicking the name of the persistent structure you reach the DDIC transaction simply through forward-navigation. There you define all the node attributes as you would do it for any other standard DDIC structure. For our sales quote demo BO we define the following fields:

- QUOTE_ID (/BOBF/S_SEPM_K_SOQ_ID or CHAR, Length 10)
- QUOTE_STATUS (/BOBF/SEPM_SQ_STATUS or CHAR, Length 2)

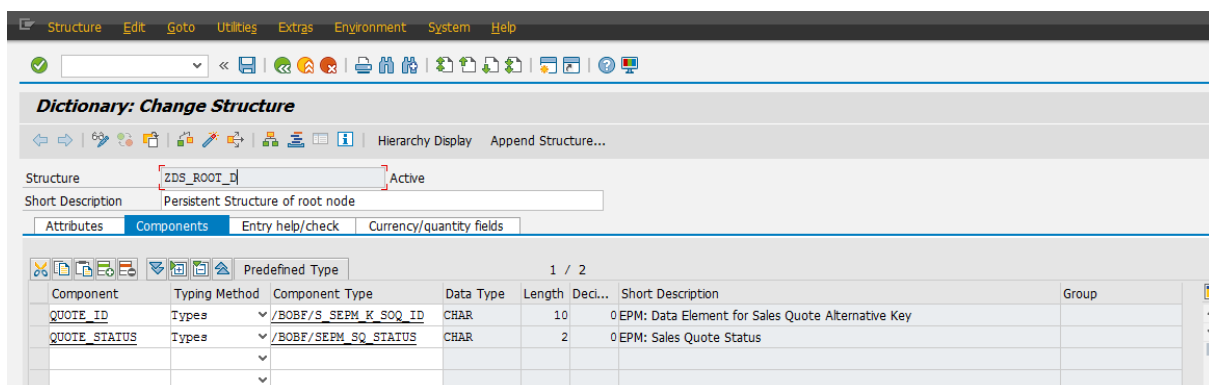


Fig. 6: Creating a persistent structure in the ABAP Dictionary (SE11)

After selecting an enhancement category for the structure (which is not relevant for our demo) and activating it, we can navigate back to the Business Object Builder transaction by selecting **Back** from the toolbar.

Back in the wizard, continue to the next step.

Choose names for root node types

In this step, you will edit the names for several DDIC database types that will be automatically generated at the end of the wizard:

Combined Structure: Combines the Persistent and the Transient Structure, and a key include, used to identify each node instance

Combined Table Type: Table type based on the Combined Structure

Database Table: Contains the data of the persistent attributes from all root node instances.

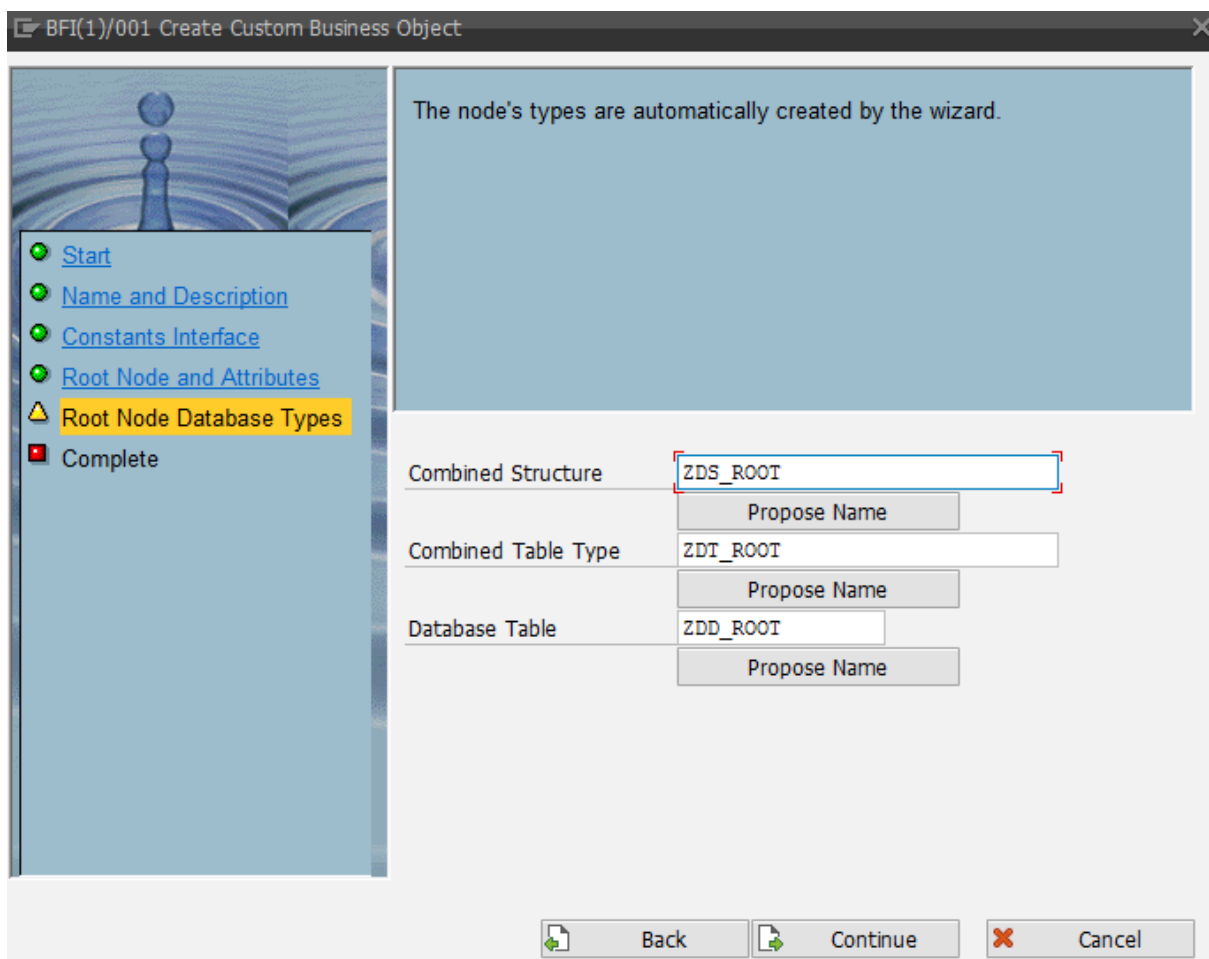


Fig. 7: DDIC artifacts that will automatically be created as data types of the root node

We now continue to the next and final step, accepting the proposed names.

Finish the BO creation procedure

To finish the BO creation, choose **Complete**. Several artifacts will now be generated in the background:

- The BO configuration
- The Constants Interface for the BO
- Combined Structure, Combined Table Type, and Database Table of the root node

You will get some popups where you have to assign the generated artifacts to a package. For our demo we can just choose \$TMP.

Result

You are now finished. The created business object is opened in the configuration view of the Business Object Builder, showing you the configuration you have created. Of course, here you would be able to change or delete the configuration, or even add further elements -- for instance, a subnode to the BO.

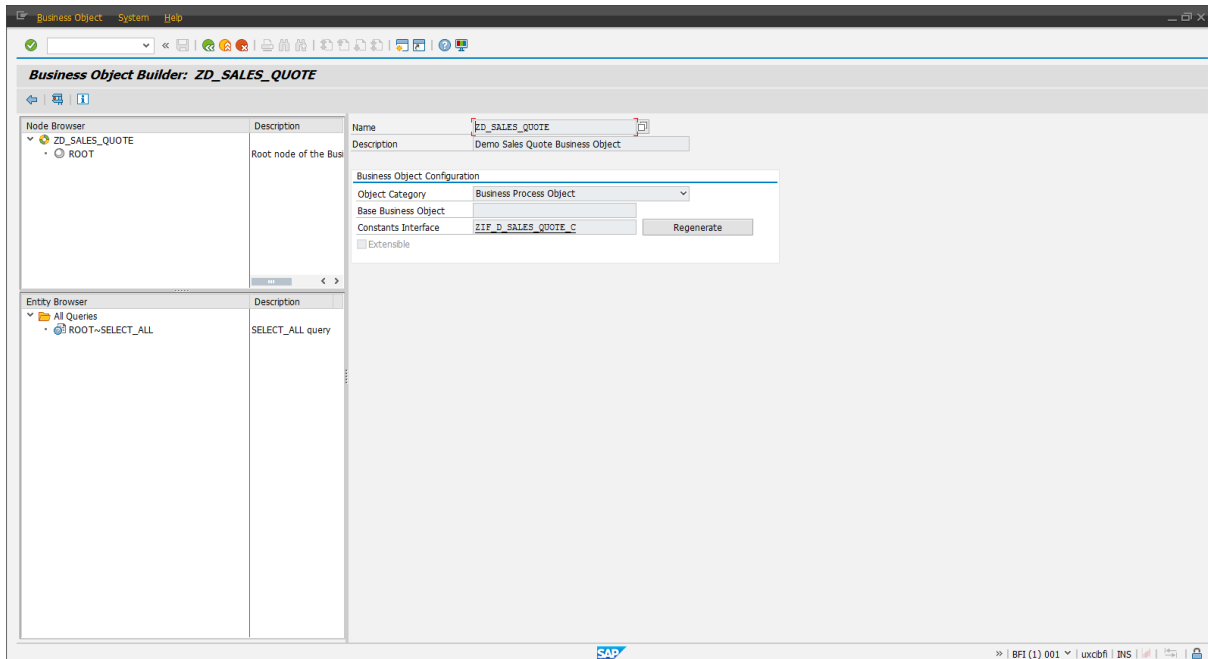


Fig. 8: BO with ROOT node and SELECT_ALL query in configuration view

At this point, the new BO is already enabled to perform some generic operations such as create, read, and update (CRUD) operations with the help of the generic services of BOPF. So that you can access persisted root node data, the system has also generated a Query called SELECT_ALL.

ADD A SUBNODE

In the second step, we want to enhance the BO structure by adding a subnode to the already existing root node.

Prerequisites

The created BO is opened in the configuration view of the Business Object Builder.

Procedure

Start Node Creation Wizard

Open the context menu of the root node in the *Node Browser* pane. Select the entry **Create Subnode**. Choose **Continue** to go to the first step.

Create subnode

Perform the wizard steps for a new subnode called ITEM, as described above for creating the root node. As the Persistent Structure you can reuse the structure /BOBF/S_SEPM_ITEMS_D. If it does not exist, simply create a new structure with the following fields:

- ITEM_POS (CHAR, Length 10)

- PRODUCT_ID (SNWD_PRODUCT_ID)
- QUANTITY (SNWD_QUANTITY with field QUANTITY_UNIT as quantity ref. field or INT4)
- QUANTITY_UNIT (SNWD_QUANTITY_UNIT or CHAR, Length 3)

Result

You have created a subnode to the root node. By default, subnodes have a cardinality of 0..n, which means - for our example - that one root node instance can have zero or many ITEM node instances. The created business object is opened in the configuration view of the Business Object Builder, showing you the new configuration.

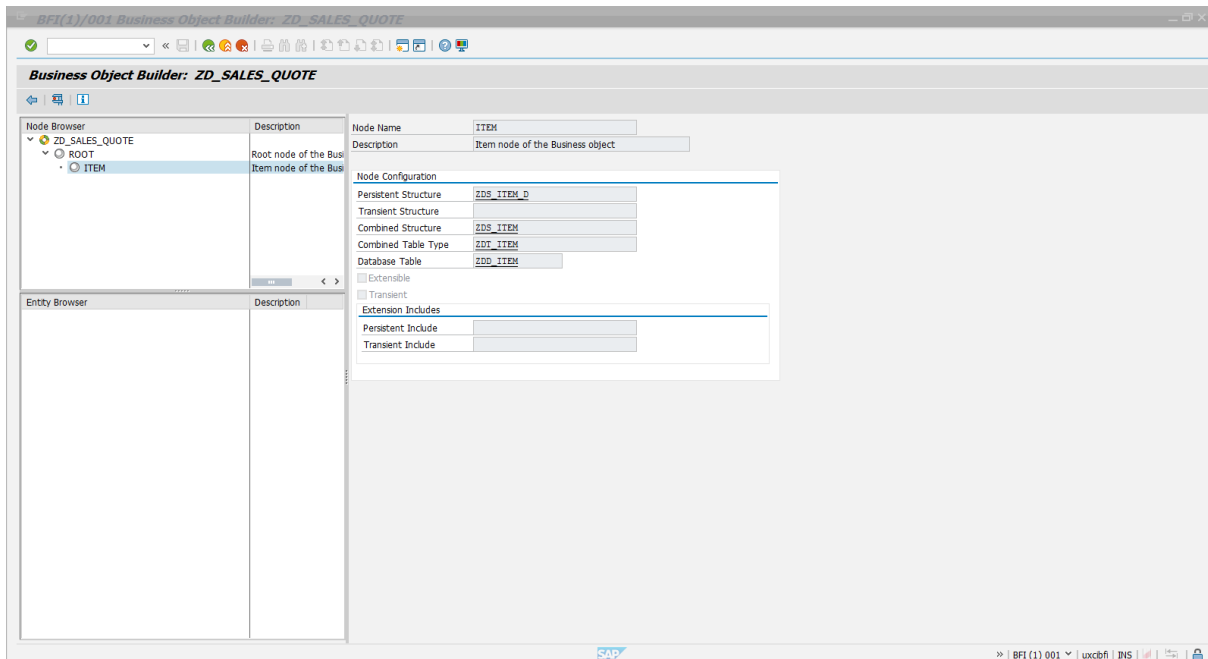


Fig. 9: BO with new ITEM node

Let's go to the next task where we want to implement some business logic.

ADD AND IMPLEMENT AN ACTION

BOPF knows several entities that can be used to define a specific behavior on the business object. An action is one of these entities, describing an operation on a certain node. With the new action we want to set the value of the QUOTE_STATUS attribute of the root node.

Prerequisites

The created BO is opened in the configuration view of the Business Object Builder.

Procedure

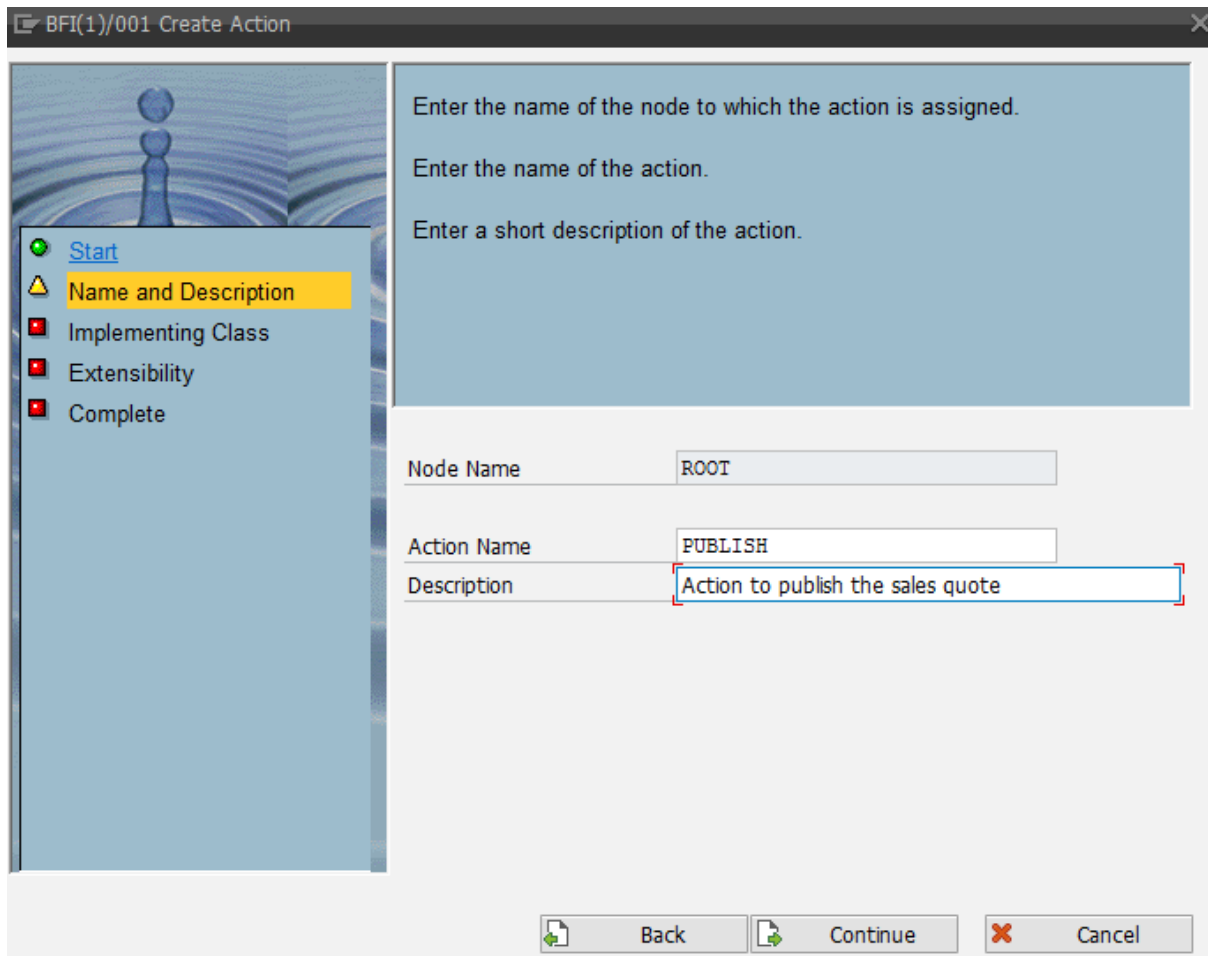
Start Action Creation Wizard

Open the context menu of the root node in the *Node Browser* pane. Select the entry **Create Action**. The *Create Action* wizard is started for guiding you through the necessary steps.

Choose **Continue**.

Define Name and Description of the Action

In this step, you have to provide the action name and you can define a short description.



BFI(1)/001 Create Action

Enter the name of the node to which the action is assigned.

Enter the name of the action.

Enter a short description of the action.

Node Name: ROOT

Action Name: PUBLISH

Description: Action to publish the sales quote

Back Continue Cancel

Fig. 10: Name of the action

The target of our action will be to set the value of the QUOTE_STATUS to 'published' in order to indicate that the sales quote was created and sent to the business partner. Thus we name the Action PUBLISH.

Continue to the next step.

Define the Implementing Class

At runtime, an action is represented by an ABAP class, implementing a certain interface. In this step you are able to define the name of that class. In addition you have to select the *Action Cardinality*. This setting indicates how many node instances the action can operate on. Due to performance reasons, we recommend that action implementations should always be designed for running on multiple instances instead of single instance only. To provide additional importing parameters for an action implementation, you have the option to specify a parameter structure for this action.

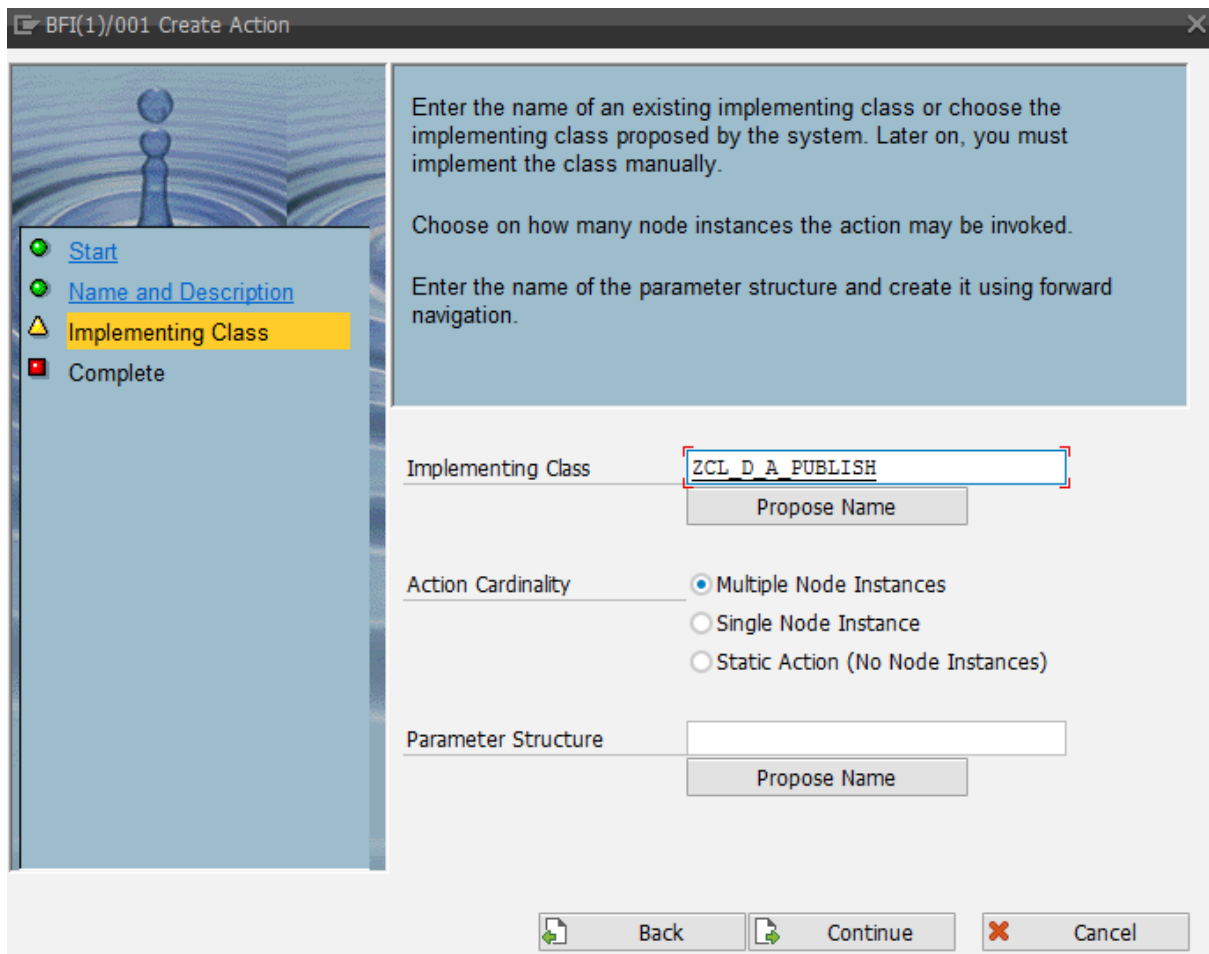


Fig. 11: Implementing the action class

For this demo we won't change the proposed name and *Action Cardinality* settings, but instead continue with the next step.

Finishing the wizard

Finish the action creation by choosing **Complete**. In the background, the system updates the BO configuration and generates the implementing class of the action.

Implement the Action

The new action is now visible in the *Entity Browser* pane of the BO configuration view. On the right, the settings of the selected action are shown. Double-click the name of the *Implementing Class* to navigate to the Class Builder. Open the empty implementation of method `/BOBF/IF_FRW_ACTION~EXECUTE` and provide the following source code.

```
METHOD /bobf/if_frw_action~execute.
  DATA lr_root          TYPE REF TO zds_root. " Combined structure
  DATA lt_changed_fields TYPE /bobf/t_frw_name.
  DATA ls_key           TYPE /bobf/s_frw_key.

  " Prepare update information
  CREATE DATA lr_root.
  lr_root->quote_status = 'P'. "Published
  APPEND zif_d_sales_quote_c=>sc_node_attribute-root-quote_status TO lt_changed_fields.
```

```
" Do update
LOOP AT it_key INTO ls_key.
  io_modify->update(
    EXPORTING
      iv_node           = zif_d_sales_quote_c=>sc_node-root
      iv_key            = ls_key-key
      is_data           = lr_root
      it_changed_fields = lt_changed_fields ).
  ENLOOP.
ENDMETHOD.
```

Hint: if you copy the source code directly into the ABAP editor, the formatting is lost. Copying it into WordPad for example preserves at least the line breaks. This format can then be copied into the ABAP Editor.

As a first step, we prepare the update information where we describe what should be changed in the affected node instances. We create a data reference for the nodes combined table type and set the attribute QUOTE_STATUS to 'P'. This is the indicator that the node instance is set to status "Published". In addition, we populate the internal table LT_CHANGED_ATTRIBUTES to describe which attribute should be updated. Like all other entity names, the attribute name, too, is available through the *Constants Interface* of the BO. (Remember that you have to use the specific *Constants Interface* of your business object that you created in the first task).

Next, the content of the importing parameter IT_KEY is processed within a LOOP. IT_KEY contains the keys of all node instances that should be processed by the action. For each node instance we call the method UPDATE from the Internal Access Object IO_MODIFY.

Note that there are also other variants of the implementation possible but for this demo I have tried to keep things simple and easy to understand.

Activate your code and navigate back to the configuration view of the Business Object Builder.

Result

You are now finished with the implementation of the first business logic. Let's go to our last task, where we want to test our business object.

TEST THE BUSINESS OBJECT

To test the functions of a Business Object, you can use the transaction Business Object Test Environment (BOBT).

Prerequisites

You have completed task 1 (Create a Business Object) and task 2 (Add an Action). Your business object is opened in the configuration view of transaction BOB.

Procedure

Start Business Object Test Environment

In the configuration view of BOB, choose the **Test** button in the toolbar.

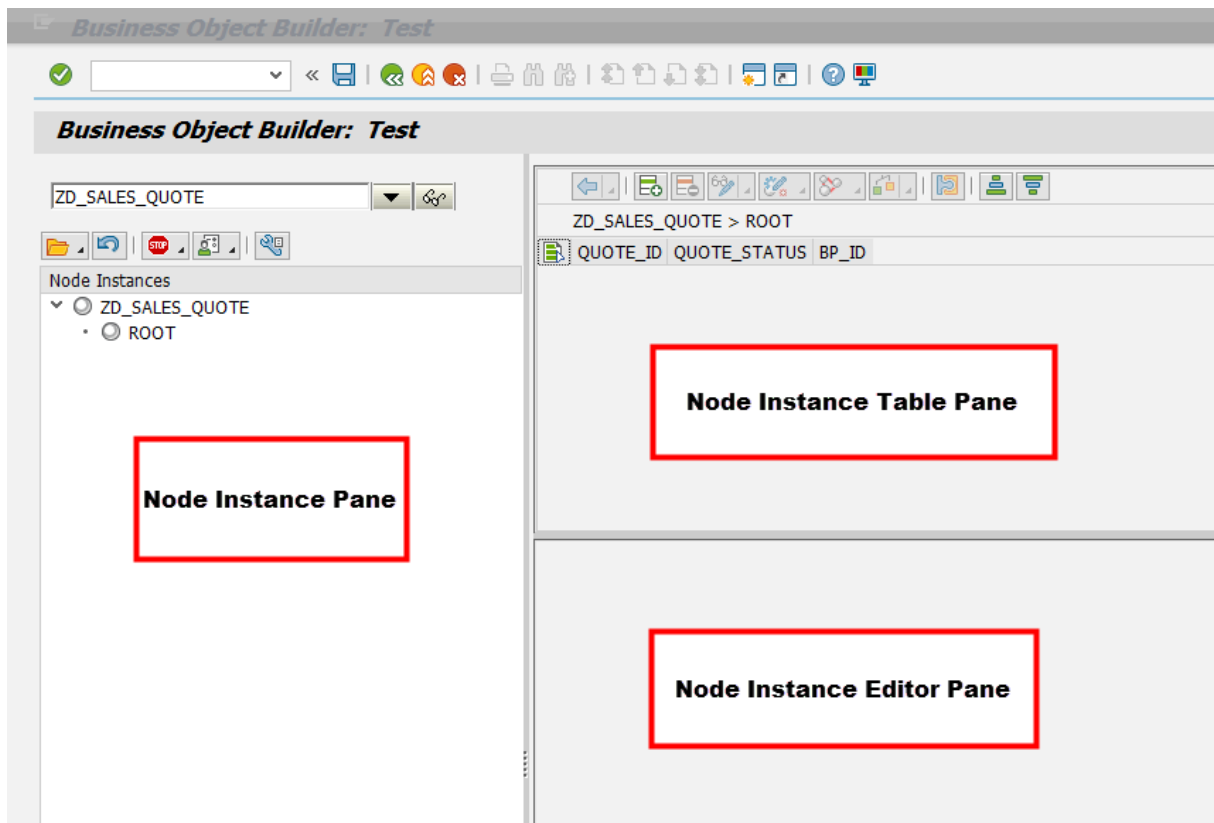


Fig. 12: Transaction BOBT

A new window is opened, showing the BO in transaction BOBT. BOBT is a generic test tool that allows you to perform any function of your Business Object without writing code. By default, BOBT consists of three panes:

- Node Instance pane: Showing all node instances of the BO in a tree along the node structure
- Node Instance Table pane: Showing all node instance of a particular node
- Node Instance Editor pane: Shows the details of a particular node instance

Create, Save and Query Node Instances

In the Node Instance Table pane, select **Add Node Instance** from the toolbar.

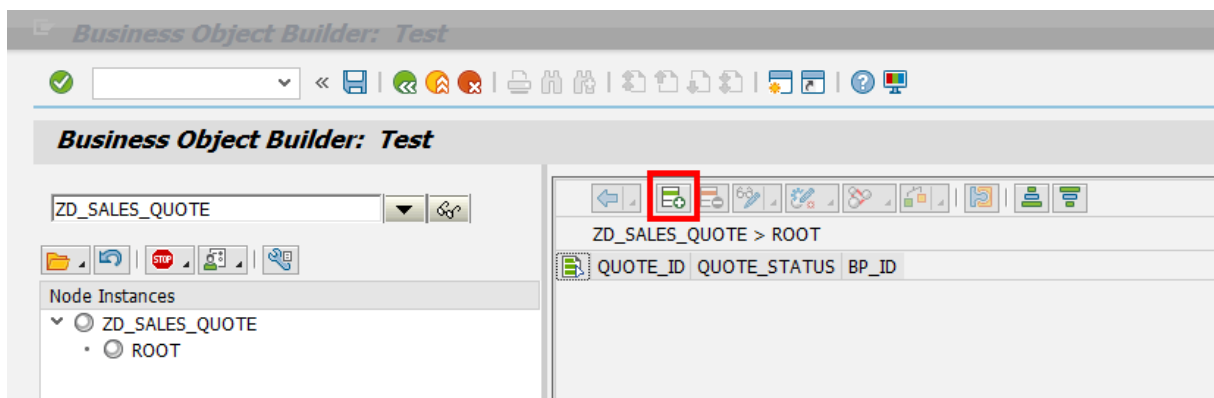


Fig. 13: Create ROOT node instance

An empty ROOT node instance appears in the *Node Instance Table* pane and the *Node Instance Editor* pane, allowing you to change the attributes by just entering values.

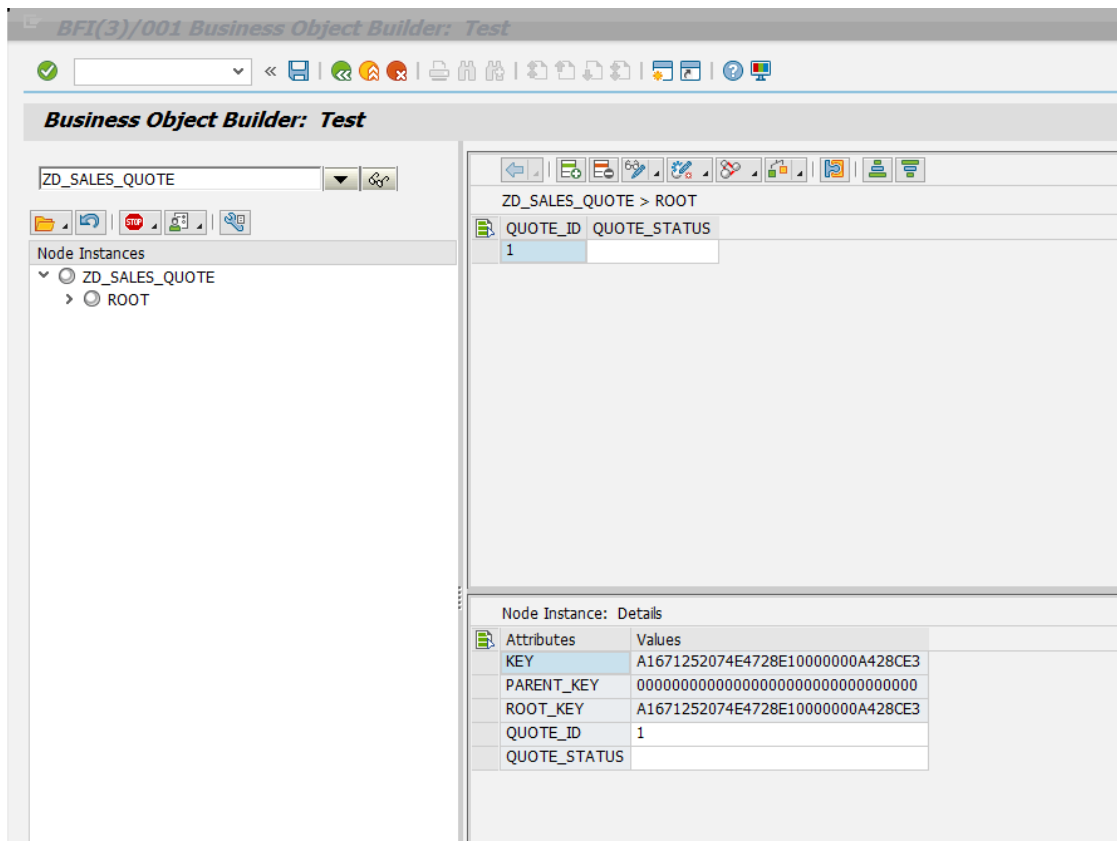


Fig. 13: Created ROOT node instance

Navigate to the ITEM node by selecting the *ITEM* association from the **Association** button.

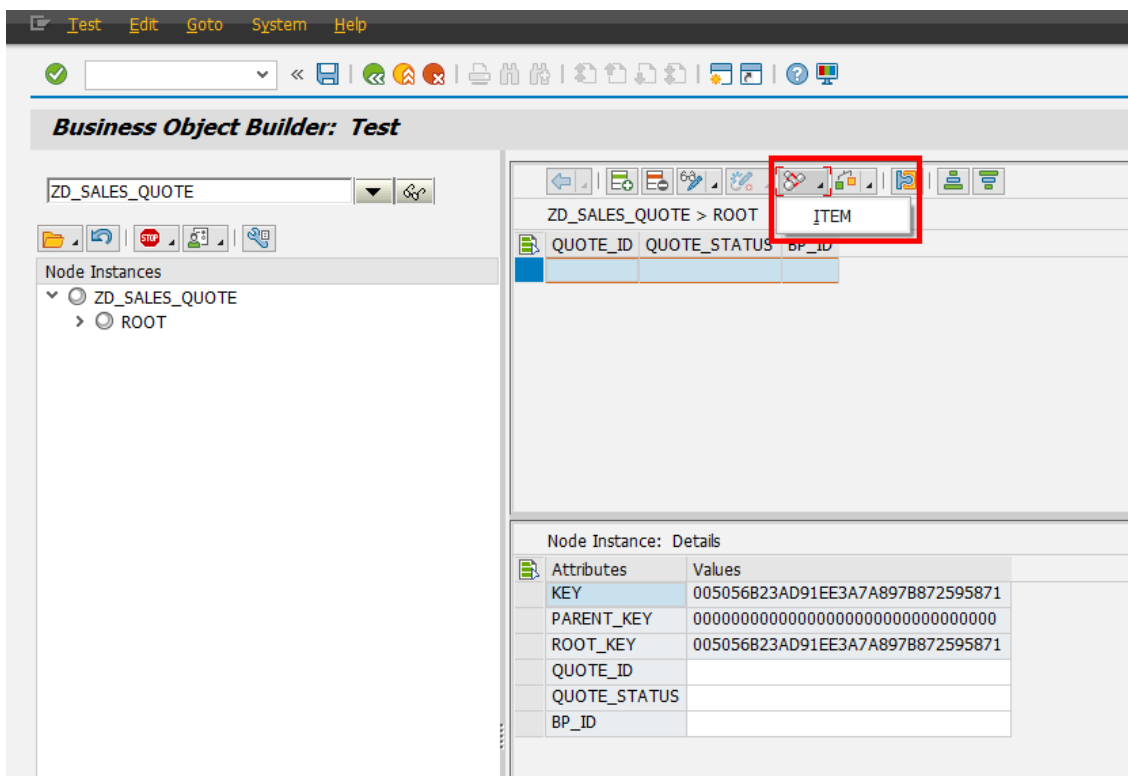


Fig. 14: Navigate to ITEM node instance

The node Instance Table pane now shows the ITEM instances that are assigned to the ROOT node instance that we have created before. Select Add Node Instance from the toolbar two times in order to create two ITEM node instances. Enter some values as you did before for the ROOT node instance.

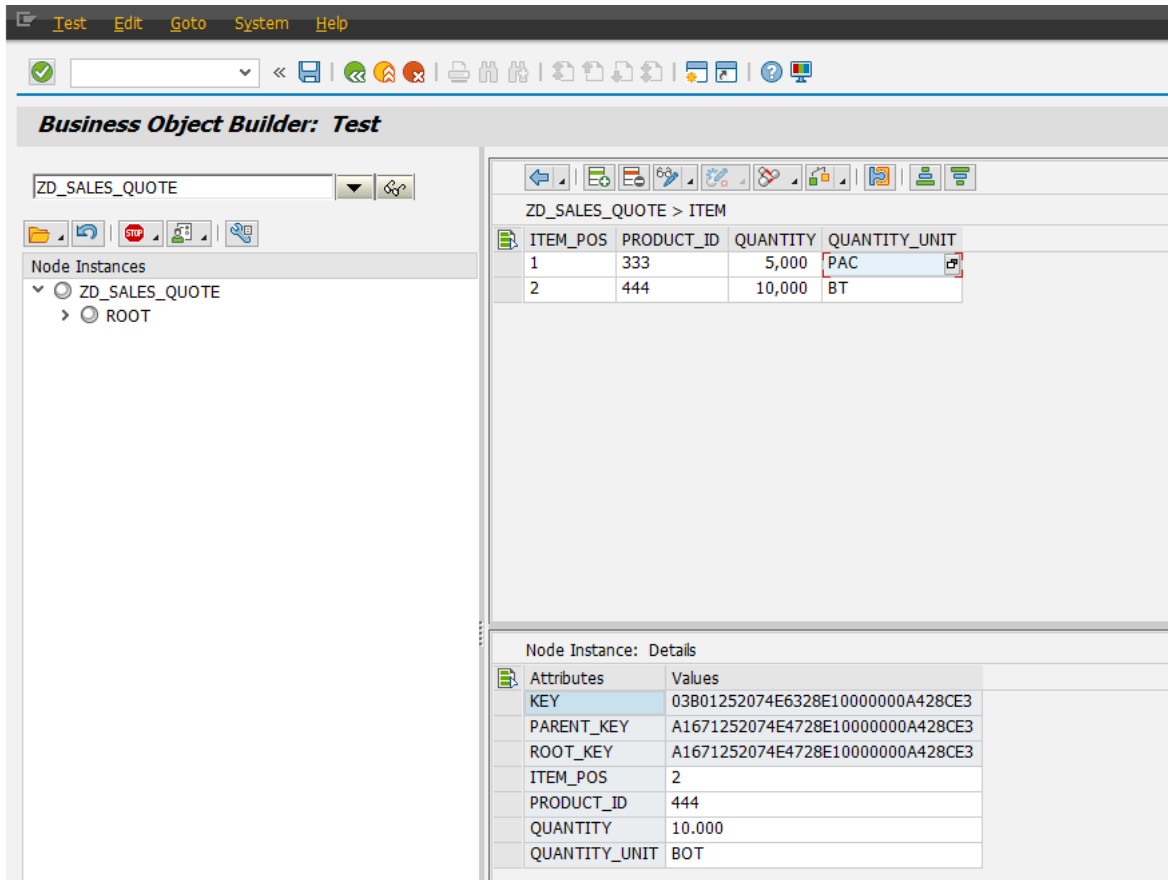


Fig. 15: Created ITEM node instances

Finally you can save your BO data by choosing **Save** from the SAP toolbar. The data is saved to the database table that is assigned to the nodes.



Fig. 15: Save BO instances

You can remove the data from the transaction and the tool BOBT by choosing **Clear Transaction** from the toolbar of the Node Instance pane.

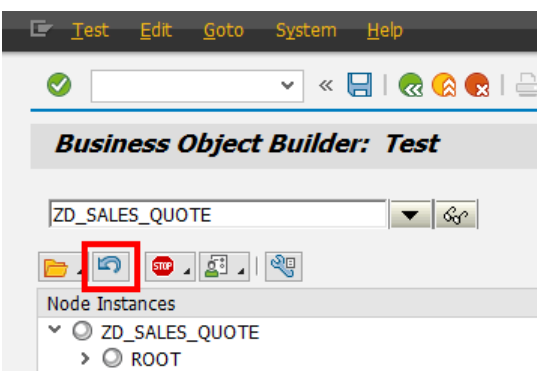


Fig. 16: Save BO instances

In our last step, let's read the data created from the database. Select the **Search** button right next to the BO name. Choose the button **Load Node Instances** and select the entry **By Query** and **ROOT – SELECT_ALL**. All ROOT node instances are loaded into the transaction and visible again in BOBT.

Execute the Action

As a final step, select the create node instance in the Node Instance Table pane and execute the action by choosing PUBLISH in the application toolbar.

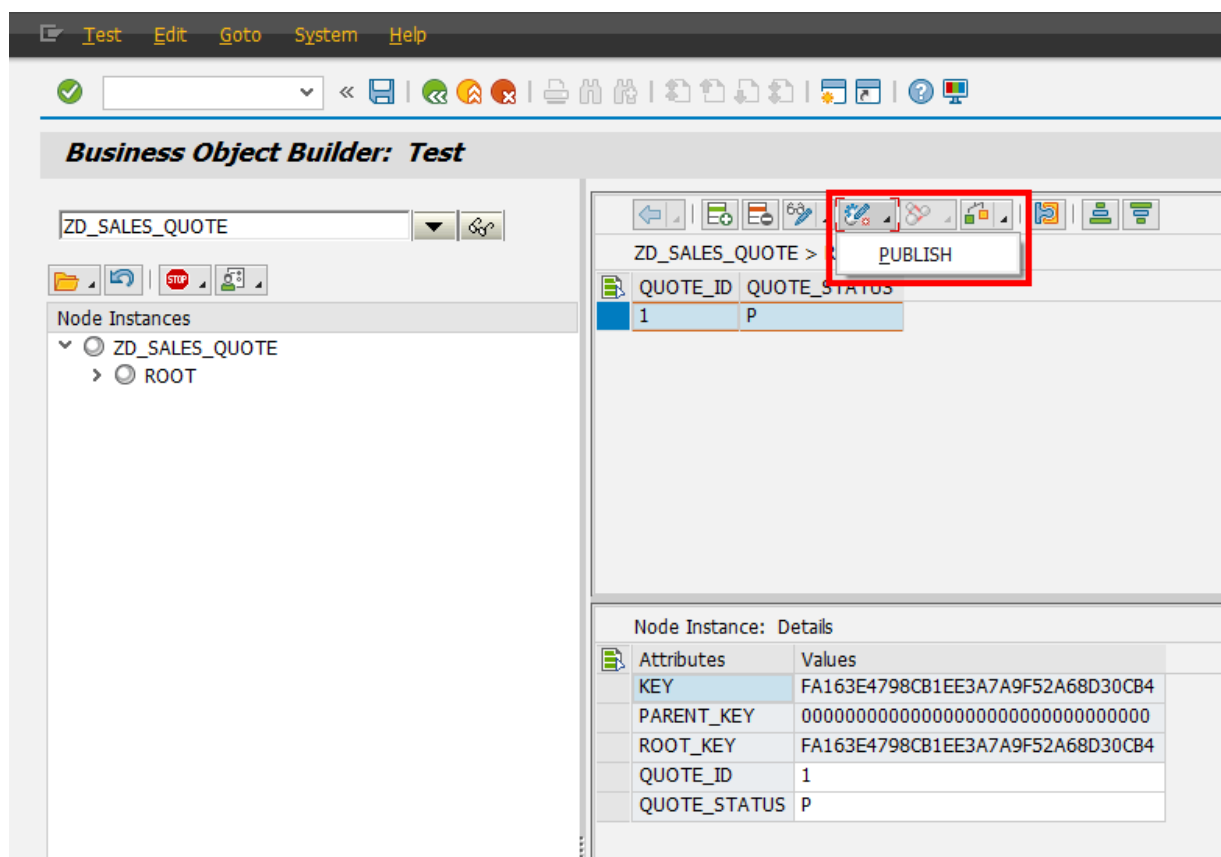


Fig. 17: Resulting ROOT node instance after action execution

As a result, you should see an update on the QUOTE_STATUS attribute in accordance with the action you implemented in the previous step.

Result

Within a few minutes we have verified that we have implemented the action correctly, without writing any test code. Of course, this does not replace an automated test, but with the help of the Business Object Test Environment you get direct feedback as to whether your Business Object works correctly or not.

We have now gone through the fundamentals of creating and testing a Business Object. I hope you are motivated to try out more things with BOPF, as there is much more that can be discovered. Stay tuned for further articles about our framework.

© 2014 SAP AG. All rights reserved.

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.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

