ESPRESSO
TUTORIALS

Dr. Boris Rubarth

Interface Programming in SAP® ABAP

- Overview of classic SAP ABAP interface techniques
- Remote Function Call (RFC) and BAPIs
- IDoc and ALE
- Using Remote Function Module (RFM) und BAPIs in SAP S/4HANA

Table of Contents

2 Using BAPIs

If you ask experienced colleagues what a BAPI is, their answers may range from “Oh, just another RFM” to “A method to access business data inside SAP solutions encapsulated in a business object”. Both statements are correct: the colleagues just see things from different angles. We will start by looking at the shorter answer: a BAPI is a remote-enabled function module.

2.1 Calling a BAPI by RFC in three minutes

A BAPI is an RFM and its name starts with BAPI_. If, for example, you want to get a list of users from the target system, your coding may look as shown in Listing 2.1:

```
REPORT  z_bapi_call.
PARAMETERS pa_dest
      TYPE rfcdes-rfcdest DEFAULT 'FIRSTDEST'.
DATA: t_user TYPE TABLE OF bapiusname,
      l_user TYPE bapiusname.

CALL FUNCTION 'BAPI_USER_GETLIST'
  DESTINATION pa_dest
  TABLES
    userlist = t_user.

LOOP AT t_user INTO l_user.
  WRITE: /, l_user-username.
ENDLOOP.
```

Listing 2.1: Simplified RFC to BAPI example

Was that all we need to know about BAPIs? Of course not … and again this introductory coding is not sufficient, so let us dig deeper.

2.2 Getting the overview on BAPIs

With the knowledge from the previous chapter on RFCs, you can already call a BAPI using RFC as the protocol, as the BAPI implementation is always an RFM. So why are you reading this chapter? BAPIs offer some very important and helpful features for your remote communication but also hold some traps—so it is worth examining some of the details. You may also be curious about what “BAPI implementation” means.

Let’s start with the name: *BAPI* is the abbreviation for *Business API*, in the sense of an interface to business data.

BAPI or BAdI

Do not confuse a BAPI with a BAdI: a *BAdI* is a *Business Add-In* for a modification-free extension of existing ABAP coding. And a BAPI may offer a BAdI—a point to extend the BAPI without modification, as we will discuss later.

If an RFM implements a BAPI, then you know how to examine the BAPI: you examine the RFM. Nevertheless, if the RFM is the **implementation**, there must be something beyond the implementation, some meta level that the RFM is implementing.

You can use the Function Builder selection screen to search for RFMs with the prefix BAPI_. (Later on, we will discuss a smarter way to search for BAPIs.) The next part of the name is the object to which the BAPI is related. Remember: the other colleague mentioned something like “data … encapsulated in a business object”.

The “object” belongs to the meta level beyond the implementation. This object is a representation of business data to which the BAPI shall offer an interface. A simple example is BAPI_USER_GETLIST: this BAPI provides a list of system users, so **User** is the object type to work on and **getlist** is the method—makes sense, doesn’t it? Figure 2.1 illustrates the business object with its methods and the related RFMs which are relevant for the RFC communication.

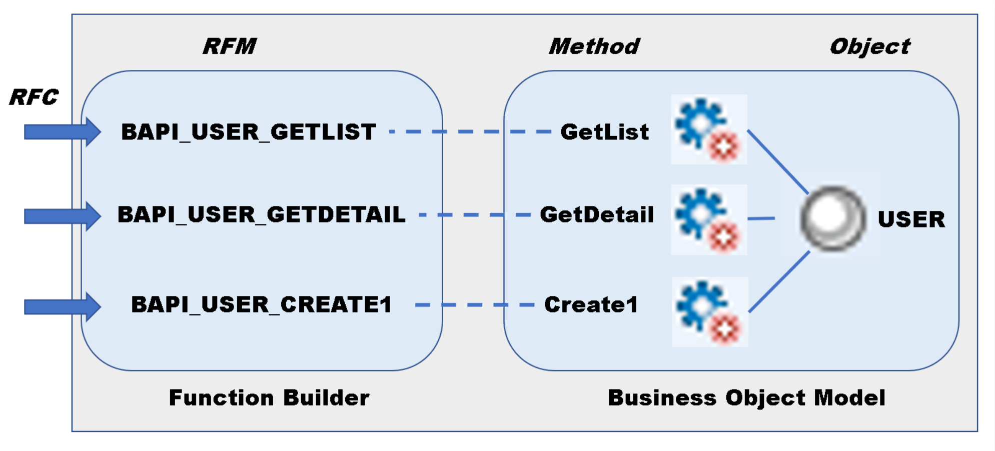


Figure 2.1: Relationship between the business object, method, and RFM

Open this BAPI (the RFM BAPI_USER_GETLIST) in the Function Builder and check the attributes: the function module is indeed remote-enabled. However, if you are looking for attributes revealing that this RFM is a BAPI, there is no clue available here, the only indication is the name.

Moreover, the source code of the BAPI-RFM is just like the source code for any other RFM. Therefore, we need another way to examine BAPIs and find out what makes them special.

Object-oriented technique or module technique

Although a BAPI relates to an object, the coding is not **object-oriented** in the sense that the object belongs to an ABAP class. No ABAP class exists for the object to which the BAPI belongs.

Nevertheless, the implementation inside the function module can use ABAP object statements, just like any other RFM implementation.

2.3 Exploring BAPIs

Instead of searching for the name of an RFM in the Function Builder, you can use the *BAPI Explorer*. This is a comfortable and well-structured tool that you can use to search for an appropriate interface for your scenario. You open the BAPI Explorer using the transaction code BAPI—it's easy to remember. Nevertheless, the BAPI Explorer initially does not show the BAPIs but rather the objects, which we call *business objects*.

The BAPI Explorer lists these business objects and their interfaces (BAPIs) by application component hierarchy (HIERARCHICAL tab) or by name (ALPHABETICAL tab).

BAPI benefit: Easy to find

You can find appropriate BAPIs for your project in the BAPI Explorer using the application component hierarchy for orientation.

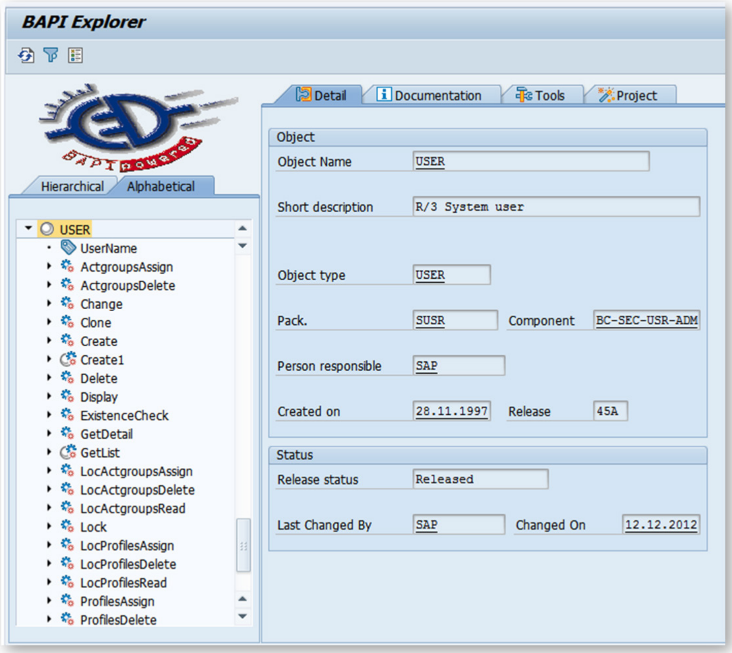
Switch to the ALPHABETICAL tab first and then scroll down to select the object *USER*. The DETAIL tab then displays details for this object, as shown in Figure 2.2.

The right-hand side of the screen offers additional tabs: DOCUMENTATION indicates that each business object, with its BAPIs, is well documented; the PROJECT tab offers a guided way to create your own business objects and BAPIs; the TOOLS tab offers several tools based on the object selected.

BAPI benefit: Well documented

You will find documentation for each BAPI. This is in contrast to RFMs, which may not be documented by SAP in each case.

Now select the method *GetList* of the object *USER*. The details reveal the name of the RFM that implements the BAPI, as shown in Figure 2.3.



Object

Object Name: USER

Short description: R/3 System user

Object type: USER

Pack.: SUSR   Component: BC-SEC-USR-ADM

Person responsible: SAP

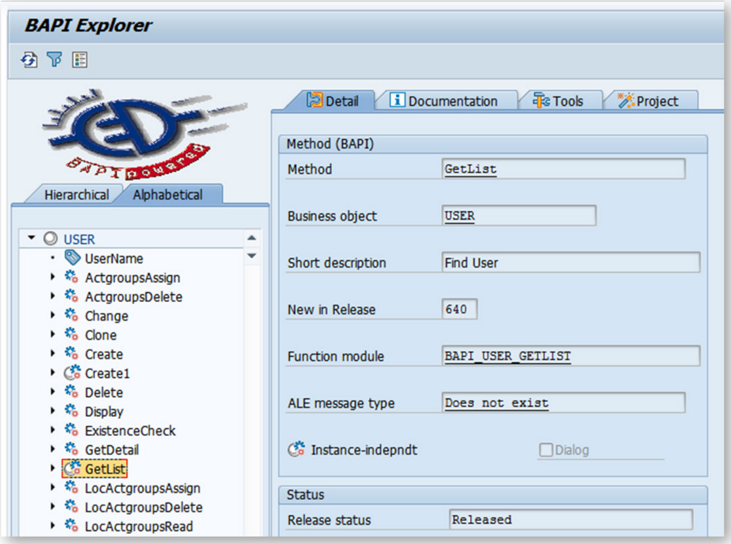
Created on: 28.11.1997   Release: 45A

Status

Release status: Released

Last Changed By: SAP   Changed On: 12.12.2012

Figure 2.2: Transaction BAPI shows details for the object USER



Method (BAPI)

Method: GetList

Business object: USER

Short description: Find User

New in Release: 640

Function module: BAPI_USER_GETLIST

ALE message type: Does not exist

Instance-indepndt   □ Dialog

Status

Release status: Released

Figure 2.3: Details for BAPI_USER_GETLIST

B Index