How to code a small SAP® server with PureBasic

by Stefan Schnell

To communicate and connect to a SAP® system, it offers a few interfaces. One of these is the Remote Function Call (RFC) interface. To get RFC access it is necessary to use the SAP® RFC SDK with its libraries. There are two versions of the SDK, the older classic version and the actual NetWeaver® version. For the classic version you find a lot of examples, but for the NetWeaver® version examples are very rare. In the following article I describe one easy example. It is how to code a small – maybe the smallest – SAP® server program. I use the programming language PureBasic and the actual version of the SAP® NetWeaver® RFC SDK.

The idea behind this article is to code the smallest SAP® server program in the whole wide world. So I reduce the code to an absolute minimum, but this is the first important step to understand how easy it is to work with SAP® NetWeaver® RFC SDK.

Preparation

Download the SAP® NetWeaver® RFC SDK from the SAP® marketplace – a passport is required. Also download SAPCAR, the SAP® archiver, to unpack the files, because the RFC SDK is packed in SAR format – SAP® archive. You need six shared libraries, also known as DLLs, from the package:

- icudt34.dll,
- icuin34.dll,
- icuuc34.dll,
- libicudecnumber.dll,
- libsapucum.dll and
- sapnwrfc.dll.

Copy them to a separate directory. It is necessary to install the Microsoft® Visual C runtime modules, you need the runtime library 2005. These are all preparations, now we take a look at the code.

Easy and small

The following code is very easy to understand. We need only four functions from the SAP® NetWeaver® RFC SDK

- RfcCreateFunctionDesc,
- RfcInstallServerFunction,
- RfcRegisterServer and
- RfcListenAndDispatch.

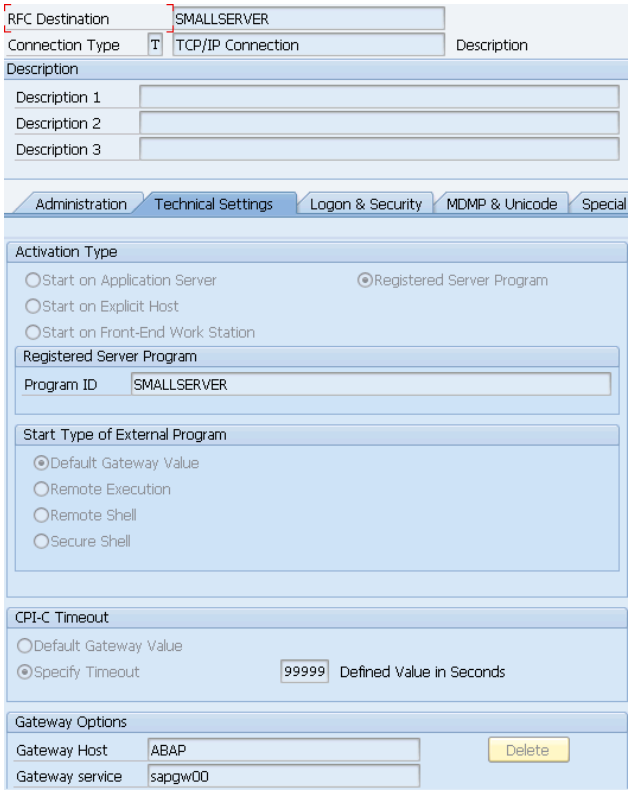
With the function RfcCreateFunctionDesc we create a description of a function which can be called from ABAP®. With the function RfcInstallServerFunction we install this description on the server and with RfcRegisterServer we register the server – that's all. Last but not least we use RfcListenAndDispatch in an endless loop and each call of the procedure ABAPCall, from an ABAP® program, executes this procedure.
The PureBasic program is a console program. Call it from the console with the arguments /g for the gateway host and /x for the gateway service. It shows the connection arguments after the start.
Here is the commented source with less than 100 lines:

```
; Begin--------------------------------------------

  ; Constants----------------------------------------
    Enumeration
      #RFC_OK
      #RFC_RETRY = 14
    EndEnumeration

  ; Structures---------------------------------------
    Structure RFC_ERROR_INFO
      code.i
      group.i
      key.s{128}
      message.s{512}
      abapMsgClass.s{21}
      abapMsgType.s{2}
      abapMsgNumber.s{4}
      abapMsgV1.s{51}
      abapMsgV2.s{51}
      abapMsgV3.s{51}
      abapMsgV4.s{51}
    EndStructure

    Structure RFC_CONNECTION_PARAMETER
      name.i
      value.i
    EndStructure

  ; Variables----------------------------------------
    Global RfcErrorInfo.RFC_ERROR_INFO
    Global hDesc.i = 0, hConn.i = 0, rc, i
    Global GatewayHost.s = #NULL$
    Global GatewayService.s = #NULL$
    Dim connParams.RFC_CONNECTION_PARAMETER(3)

  ; Procedure ABAPCall-------------------------------
    Procedure.i ABAPCall(rfcHandle.i, funcHandle.i,
*errorInfo)
      PrintN("ABAPCall received")
      ProcedureReturn #RFC_OK
    EndProcedure

  ; Main---------------------------------------------
    If OpenLibrary(0, "sapnwrfc.dll") And OpenConsole()

      For i = 0 To CountProgramParameters() - 1
        Select ProgramParameter(i)
          ; GatewayHost-----------------------------
          Case "/g", "/G"
            GatewayHost = ProgramParameter(i + 1)
          ; GatewayService--------------------------
          Case "/x", "/X"
            GatewayService = ProgramParameter(i + 1)
        EndSelect
      Next

      PrintN("Example for a small SAP server")
      Print("Host: " + GatewayHost)
      PrintN(" / Service: " + GatewayService)
      PrintN("Break execution with Ctrl+C")

      hDesc = CallFunction(0, "RfcCreateFunctionDesc",
@"ABAPCall", RfcErrorInfo)

      If hDesc And GatewayHost <> "" And GatewayService
<> ""

        connParams(0)\name = @"program_id"
        connParams(0)\value = @"SMALLSERVER"
        connParams(1)\name = @"gwhost"
        connParams(1)\value = @GatewayHost
```

```
        connParams(2)\name = @"gwserv"
        connParams(2)\value = @GatewayService

        rc = CallFunction(0,
"RfcInstallServerFunction", #Null, hDesc, @ABAPCall(),
RfcErrorInfo)

        If rc = #RFC_OK
          hConn = CallFunction(0, "RfcRegisterServer",
connParams(), 3, RfcErrorInfo)
          If hConn
            While rc =#RFC_OK Or rc = #RFC_RETRY
              rc = CallFunction(0,
"RfcListenAndDispatch", hConn, 32, RfcErrorInfo)
              Select rc
                Case #RFC_OK

                Case #RFC_RETRY

              EndSelect
            Wend
          EndIf
        EndIf

      EndIf

      CloseLibrary(0)
      CloseConsole()
    EndIf

; End-------------------------------------------------
End
```

The Gateway and Customizing

To know the gateway service (gwserv) and the host (gwhost) use transaction code (TAC) SMGW and the menu Goto > Parameters > Display, look at the attribute entries gateway hostname and gateway service. Use these attributes to call the program. But before you call the program, insert in the file Windows\system32\drivers\etc\services the entry sapgw00 with 3300/tcp for sapgw00, or e.g. for sapgw99 3399/tcp. You must also customize with TAC SM59 the RFC destination SMALLSERVER.



Use the Function from ABAP®

With one line code now we can use the server function from ABAP®:

```
"-Begin---------------------------------------------
  Report zSmallServerTest.

    Call Function 'ABAPCall' Destination 'SMALLSERVER'.

"-End-----------------------------------------------
```

Summery

On the one hand the SAP® server program is very small. The commented source has 96 lines code and the compiled console executable has a size of 9.728 bytes. On the other hand this example shows how easy it is, to code a SAP® server. With these possibilities you can code your functions in PureBasic and use them in ABAP®.

Links

- service.sap.com/connectors
- www.purebasic.com
- www.stschnell.de

Trademarks

- SAP, NetWeaver and ABAP are registered trademarks of SAP AG, Germany
- Microsoft and Visual C are registered trademarks of Microsoft Corporation, USA
- PureBasic is the property of Frédéric Laboureur, Fantaisie Software, France