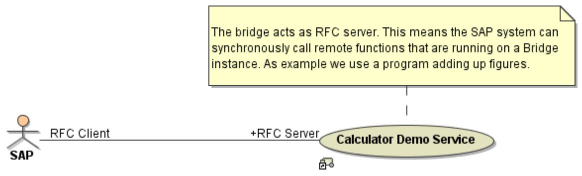
RFC Service

This is example shows how SAP calls remotely a function module running on the Bridge. This function module takes two numbers and returns their sum.

Figure: RFC Service Use Case



The bridge acts as RFC server. This means the SAP system can synchronously call remote functions that are running on a Bridge instance. As example we use a program adding up figures.
RFC Client
+RFC Server
Calculator Demo Service
SAP

Example File (Builder project Add-ons/SAP):

<your example path>\Add-ons\SAP\uml\sapRFCServer.xml

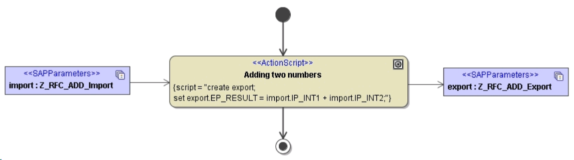
Related Pages:

- tRFC Service
- Frontend Components
- RFC Arguments
- SAP - ABAP Types Mappings

Implementing RFC Operations

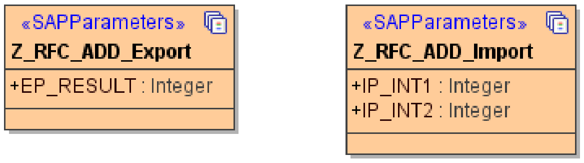
The following example just adds two numbers. The only feature of this activity diagram being specific to SAP are the import and export parameters and the fact that is assigned to an operation of a <<SAPRFCModule>> class. These properties make this activity diagram callable from SAP systems. The only constraints on such activity diagrams are that their interface is restricted to **import**, **export**, **changing** and **tables** parameters (see RFC Arguments).

Figure: Implementation of SAP RFC operation



<<SAPParameters>>
import : Z_RFC_ADD_Import
<<ActionScript>>
Adding two numbers
{script = "create export;
set export.EP_RESULT = import.IP_INT1 + import.IP_INT2;"}
<<SAPParameters>>
export : Z_RFC_ADD_Export

The following classes give the import and export arguments:



Defining RFC input- and output arguments are the same as for clients and are explained in chapter RFC Arguments.

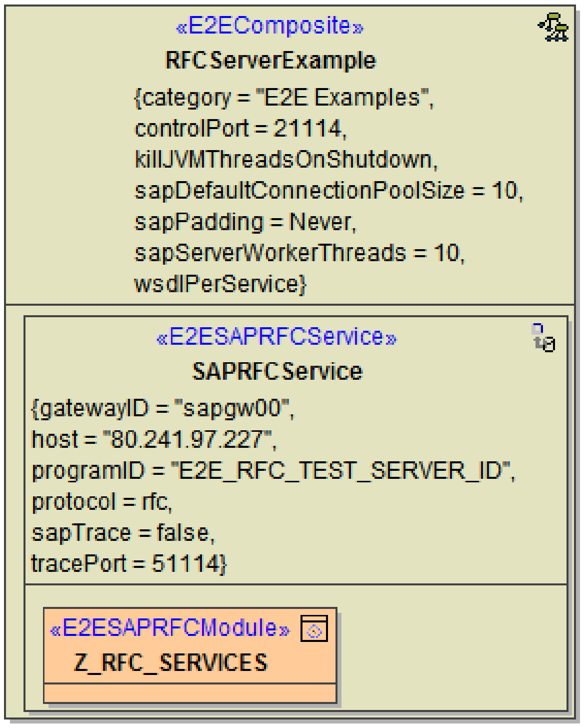
RFC Service Components

In order to build a Bridge SAP RFC Server, the following components must be defined:

- The xUML composite **RFCServerExample** holding all services you want to deploy.
- Within the composite there are one or more SAP RFC services having the stereotype <<E2ESAPRFCService>>. In the current simple example it is the **SAPRFCService**.
- Each service holds one or more classes with stereotype <<E2ESAPRFCModule>> that define the interface of the service. The activity diagrams assigned to the class operations define the implementation of the operations. In our example, the activity diagram **Z_RFC_ADD** implements the operation **Z_RFC_ADD** of the class. **Z_RFC_SERVICES**. **Z_RFC_ADD** is also the function name used within SAP ABAP programs to call the operation **Z_RFC_ADD**. See also the comment attached to the **Z_RFC_ADD** operation.

Builder 6 Modeling RFC services follows the same rules as modeling tRFC servers with two exceptions: the protocol tagged value must be **rfc** and there exists no <<SAPTRFCCallback>> interface.

Figure: RFC Service Components



The **RFCService** has the following tagged values:

Tagged Value	Description	Mandatory /Optional
<<E2EComposite>>		
sapDefaultConnectionPoolSize	Default capacity of a single SAP connection pool (Bridge acting as a SAP client). If undefined, a default of 10 connections will be applied. Each distinct connection to a SAP system has its own pool. Connections are distinguished by the set of connection parameters (connection string). You can override the connection pool size for a specific connection on the corresponding SAP alias. On using dynamic SAP access, the default connection pool size is used.	optional
sapPadding	Service-wide setting for SAP values padding. This setting will be applied to all IDoc and SAP adapters within the service. It is not recommended to use **Mixed** padding. This option is only available for reasons of backwards compatibility. Mixed padding is default for older services that have been compiled before the implementation of this tagged value, whereas **Never** is default, if no SAP padding is specified.	optional
sapServerWorkerThreads	Number of parallel request (workers) the Bridge (acting as an RFC server) can process. If this value is undefined, the Bridge will only process one request at a time (equivalent to **sapServerWorkerThreads**=1). Each active worker requires one license slot (concurrent connection). For more information on licensing and concurrent connections, refer to License for Running xUML Services.	optional
<<E2ESAPRFCService>>		
host	Optional gateway host name. Default is **localhost**.	optional

gatewayID	The port number of the SAP gateway.	mandatory
programID	The programID to which the service is registered on the gateway.	mandatory
protocol	Must be **trfc** when using the tRFC protocol. Must be **rfc** when using the RFC protocol.	mandatory
sapTrace	The effect of this flag being true is two fold: First, the SAP RFC libraries will write trace file information (**.trc**) into the directory the configuration has been deployed to. Second, by using the SAP transaction ***SMGW** (SAP gateway monitor) we can monitor the dataflow from and to the gateway the server is registered on.	optional
tracePort	The same operations that are called by the SAP system can also be called by SOAP test tool. However, the SOAP test tool requires an HTTP TCP/IP port. This port can be defined in the **tracePort** tagged value. If this value is not set, the trace port is given by **controlPort** + 50000.	optional

On the composite, you can also set a service-wide **SAP value padding**: Never, Always and Mixed. See Frontend Components for more information.

Example of an ABAP Function

The following example ABAP function (Z_CALL_RFC_SERVER) calls the **Z_RFC_ADD** Bridge operation by using the following ABAP code:

```
FUNCTION Z_CALL_RFC_SERVER.
 *"----------------------------------------------------------------------
 *"Local interface:
 *"  IMPORTING
 *"     VALUE(IP_VALUE1) TYPE  I DEFAULT 0
 *"     VALUE(IP_VALUE2) TYPE  I DEFAULT 0
 *"  EXPORTING
 *"     VALUE(EP_SUM) TYPE  I
 *"----------------------------------------------------------------------

 DATA: EP_RESULT TYPE i VALUE 0.

 CALL FUNCTION 'Z_RFC_ADD'
 DESTINATION 'E2E_RFC_TEST_SERVER2'
 EXPORTING IP_INT1 = IP_VALUE1
           IP_INT2 = IP_VALUE2
 IMPORTING EP_RESULT = EP_SUM.

 ENDFUNCTION
```