

RFC Arguments Service



This page explains the **SAP Adapter** in Bridge context. If you were looking for the same information regarding the **PAS Designer**, refer to **SAP Adapter** in the Designer guide.

The adapter interface follows the SAP ABAP conventions. Each SAP ABAP function has four parameter sections: import, export, changing and tables as shown for example in figure [Export parameters in SAP](#). In UML, these parameters are mapped to the input and output parameters of the **<<SAPRFCAdapter>>** (see for example figure [Calling Z_TEST_TYPES](#)) or RFC operations (see figure [Implementation of SAP RFC operation](#)):

Name	Type	Direction	Description
connectionString	String	in	Supplies the connection string (optional).
import	Any	in	The class specifying the type of this parameter must have stereotype <<SAP parameters>> . The attributes and associations of this class correspond to the parameters given by the import section of the ABAP function declaration – see figure Export parameters in SAP .
export	Any	out	The class specifying the type of this parameter must have stereotype <<SAP parameters>> . The attributes and associations of this class correspond to the parameters given by the export section of the ABAP function declaration
changing	Any	in/out	The class specifying the type of this parameter must have stereotype <<SAP parameters>> . The attributes and associations of this class correspond to the parameters given by the changing section of the ABAP function declaration
tables	Any	in/out	The class specifying the type of this parameter must have the <<SAP Tables>> . The attributes and associations of this class correspond to the parameters given the tables section of the ABAP function declaration.

On this Page:

- Parameters
- Tables
- Valid Native/Internal Type Pairs

Related Pages:

- [SAP - ABAP Types Mappings](#)

Parameters

When calling for example the Z_TEST_TYPES function we have a set of import (**input**) and export (**output**) parameters. These sets correspond to the attributes in the **Export** and **Import** classes. Each attribute can have the following tagged values:

Tagged Value	Description	Mandatory
nativeType	native ABAP type. For allowed types and valid internal/native type combinations see appendix beneath.	mandatory
internalLength	except for FLT, INT; DATS, TIMS): parameter length as given in the ABAP dictionary.	mandatory
decimals	number of decimals	mandatory for native type DEC

In any case, the upper multiplicity of the attributes or associations must NOT be greater than one. Otherwise, the classes cannot be mapped to SAP parameters and SAP tables should be used instead.

Figure: Export parameters in SAP

Parameter name	Type spec.	Reference type	Pass val.	Short text
EP_INT	TYPE	I	<input checked="" type="checkbox"/>	Output test for integer.
EP_DEC	TYPE	Z_DEC_TEST1	<input checked="" type="checkbox"/>	Output test for packed numbers.
EP_FLT	TYPE	FLOAT	<input checked="" type="checkbox"/>	Output test for floats.
EP_CHAR	TYPE	Z_CHAR_TEST1	<input checked="" type="checkbox"/>	Output test for chars.
EP_DAT	TYPE	Z_DATE_TEST1	<input checked="" type="checkbox"/>	Output test for data.
EP_TIM	TYPE	Z_TIME_TEST1	<input checked="" type="checkbox"/>	Output test for time.
EP_RAW	TYPE	Z_RAW_TEST1	<input checked="" type="checkbox"/>	Output test for raw byte streams.
EP_SMALL_DEC	TYPE	Z_SMALL_DEC_TEST1	<input checked="" type="checkbox"/>	Output test for a small decimal.
EP_NUNC	TYPE	Z_NUNC_TEST	<input checked="" type="checkbox"/>	Output test for numerical character
EP_LCHR	TYPE	Z_LCHR_TEST	<input checked="" type="checkbox"/>	Output test for long char.

Figure: UML classes describing the interface of Z_TEST_TYPES

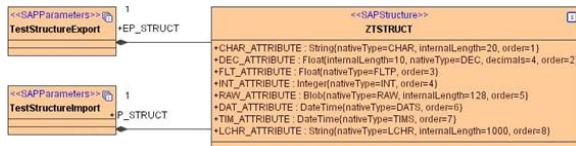


All examples so far handled simple type parameters only. However, it is also possible to assign SAP structure to SAP parameters, as the following example shows:

Function module: Z_TEST_STRUCTURE Active						
Attributes Import Export Changing Tables Exceptions Source code						
Parameter name	Type	Reference type	Default value	Opt.	Pa...	Short text
IP_STRUCT	LIKE	ZTSTRUCT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Input test structure.

Structure: ZTSTRUCT Active						
Attributes Components Entry helpcheck Currency/quantity fields						
Component	Component type	DType	Length	Dec.p.	Short text	
CHAR_ATTRIBUTE	Z_CHAR_TEST1	CHAR	20		Data element to test char handling by the E2E Bridge.	
DEC_ATTRIBUTE	Z_DEC_TEST1	DEC	10		Data Element to test packed numbers.	
FLT_ATTRIBUTE	Z_FLT_TEST1	FLTP	16		Field of type FLTP	
INT_ATTRIBUTE	Z_INT_TEST1	INT4	10		Natural number	
RAW_ATTRIBUTE	Z_RAW_TEST1	RAW	128		Data element to test raw byte stream handling by the Bridge.	
DAT_ATTRIBUTE	Z_DATE_TEST1	DATS	8		Data element to date handling by the E2E Bridge.	
TIM_ATTRIBUTE	Z_TIME_TEST1	TIMS	6		Data element to test time field handling by the E2E Bridge.	
LCHR_ATTRIBUTE	Z_LCHR_TEST1	LCHR	1000		Test LCHR domain.	

The above example shows the function Z_TEST_STRUCTURE. This operation is found in the class **Z_TEST_STRUCTURE** in figure UML classes describing the interface of Z_TEST_TYPES above. This operation as one export and one import parameter named **EP_STRUCT** respectively **IP_STRUCT**. Both parameters are of type **ZTSTRUCT** having the stereotype **<<SAPStructure>>**:



Tables

When calling for example the IDOC_INBOUND_ASYNCHRONOUS function we get or send two SAP tables :

- IDOC_CONTR_REC_40: containing EDI_DC40 structures
- IDOC_DATA_REC_40: containing EDI_DD40 structures

Figure: Configuring Tables parameters in SAP

Function module: IDOC_INBOUND_ASYNCHRONOUS Active						
Attributes Import Export Changing Tables Exceptions Source code						
Parameter name	Type spec	Reference type	Optional	Short text	Long text	
IDOC_CONTR_REC_40	LIKE	EDI_DC40	<input type="checkbox"/>			
IDOC_DATA_REC_40	LIKE	EDI_DD40	<input type="checkbox"/>			

Figure: SAP table record example: EDI_DD40

Dictionary: Display Structure

Structure: EDI_DD40 Active

Short text: IDoc Data Record for Interface to External System

Attributes Components Entry help/check Currency/quantity fields

Search help Built-in type

Component	Component type	DType	Length	Dec.p.	Short text
SEGNAM	EDI4SEGNAM	CHAR	30	0	Segment (external name)
MANDT	EDI4MANDT	CLNT	3	0	Client
DOCNUM	EDI4DOCNUM	CHAR	16	0	Doc number
SEGNUM	EDI4SEGNUM	CHAR	6	0	Segment number
PSGNUM	EDI4PSGNUM	NUMC	6	0	Number of superior parent segment
HLEVEL	EDI4HLEVEL	CHAR	2	0	Hierarchy level of SAP segment
SDATA	EDI4SDATA	LCHR	1000	0	Application data

All SAP tables consist of structures, such as EDI_DD40. Thus, an SAP table parameter can be modeled as an association to complex types of stereotype `<<SAPStructure>>` having a multiplicity greater than one. For example, the SAP table IDOC_DATA_REC40 is modeled as association to the **EDI_DD40** class corresponding to the EDI_DD40 SAP structure having the upper multiplicity of '*'. The association name equals the SAP table parameter name: IDOC_DATA_REC40.

However, sometimes it is convenient to map records to flat, simple types like String. This means, if we model the table as array of simple types like for instance strings, the Server will map the record to a simple type. The latter case we call unstructured tables, the first case structured tables. The following class diagram shows examples of both variants.

In any case, the upper multiplicity of the attributes or associations must be greater than one. Otherwise, these classes cannot be mapped to SAP tables.

Figure : Unstructured SAP RFC tables

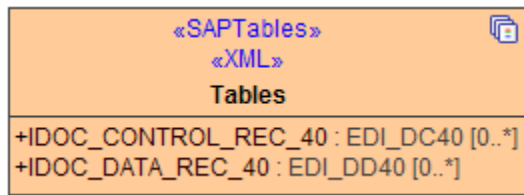
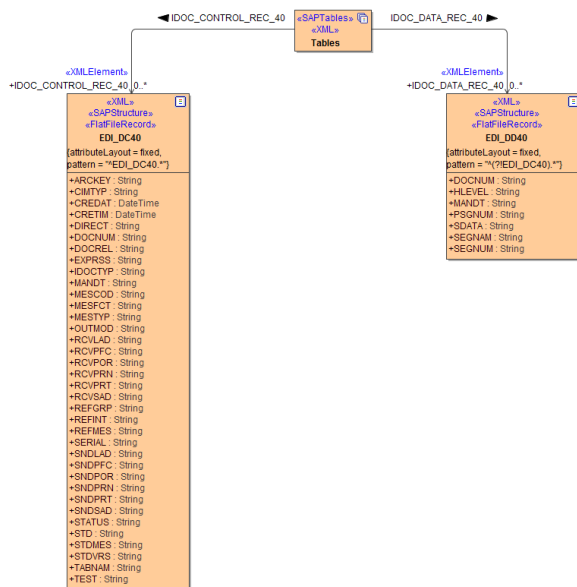


Figure: Structured SAP RFC tables



Valid Native/Internal Type Pairs

The following native SAP types are yet supported . Each native type is mapped to an internal type. Because SAP types are sometimes restricted in their length and number of decimals, we use tagged values to transport this meta information. For details of the native type definitions, please refer to the SAP documentation.

Native Type	Internal Type	Internal Length	Decimals
CHAR	String	required	n/a
LCHR	String	required	n/a
CLNT	String	required	n/a
CUKY	String	required	n/a
LANG	String	required	n/a
UNIT	String	required	n/a
ACCP	String	required	n/a
NUMC	String	required	n/a
FLTP	Float	8 bytes	n/a
DEC	Float	required	optional
QUAN	Float	required	optional
CURR	Float	required	optional
DATS	DateTime	8 bytes	n/a
TIMS	DateTime	6 bytes	n/a
RAW	Blob	required	n/a
RAWSTRING	Blob	required	n/a
INT	Integer	4 bytes	n/a
INT4	Integer	4 bytes	n/a
INT1	Integer	1 byte	n/a
INT1	Integer	2 bytes	n/a