RFC Arguments Service

On this Page:

- Parameters
- Tables
- Valid Native/Internal Type Pairs

Related Pages:

- SAP - ABAP Types Mappings

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 <<SAPParameters>>. 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 <<SAPParameters>>. 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 <<SAPParameters>>. 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 <<SAPTables>>. The attributes and associations of this class correspond to the parameters given the **tables** section of the ABAP function declaration.

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

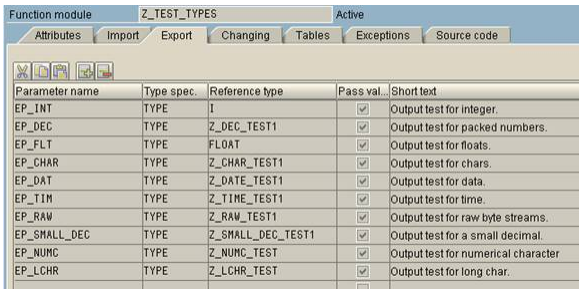
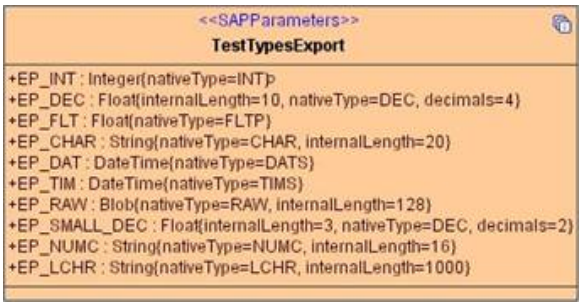
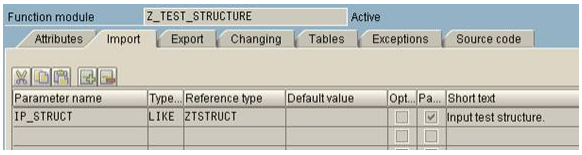
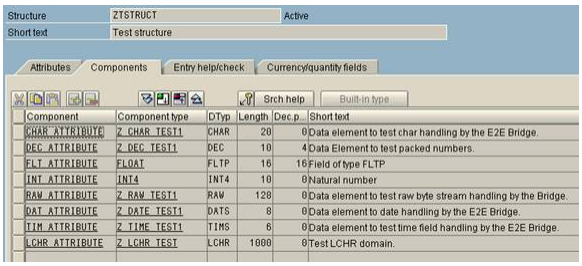


Figure: UML classes describing the interface of Z_TEST_TYPES

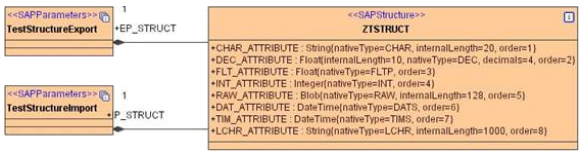


All examples so far handled simple type parameters only. However, it is also possible to assign SAP structure to SAP parameters, as the following example shows:





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

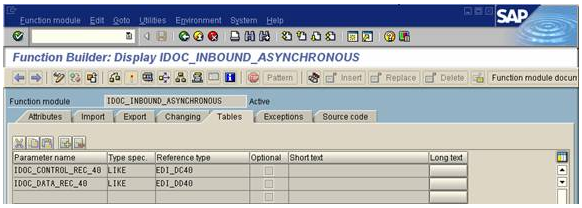
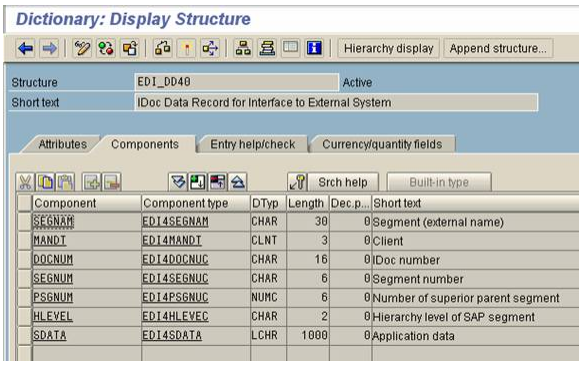


Figure: SAP table record example: EDI_DD40



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

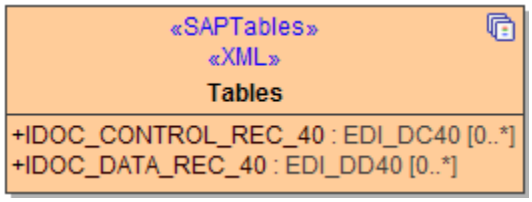
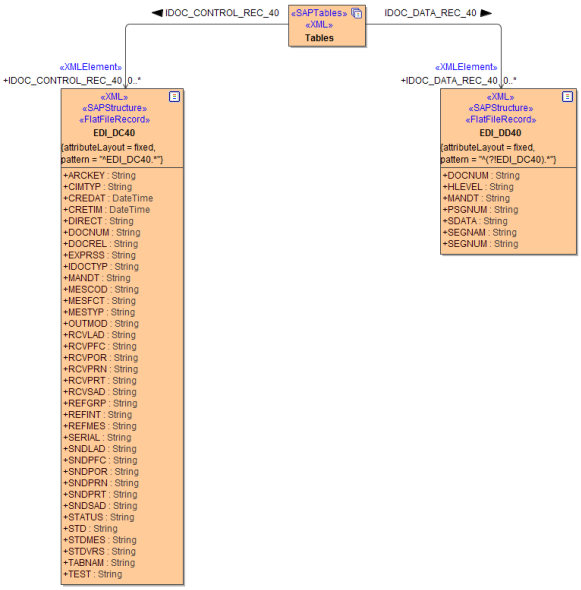


Figure: Structured SAP RFC tables



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