

Monitoring IDocs with the SAP Application Interface Framework



Applies to:

SAP Application Interface Framework 2.0

Summary

The SAP Application Interface Framework provides four different possibilities to monitor IDocs. The document provides an overview about the implementations necessary for the different scenarios. Furthermore, it provides a general overview about the customizing and configuration steps required for each scenario.

Author: Verena Wörner

Company: SAP AG

Created on: 25 October 2012

Author Bio

Verena Wörner is a developer for SAP Custom Development. She is a member of the development team for the SAP Application Interface Framework.

Table of Contents

Introduction	3
Generate IDoc Structure and Interface Definition.....	3
Scenario 1: Monitoring Existing IDocs in Monitoring and Error Handling.....	4
Scenario 2: Process IDoc Using AIF, Call IDoc Function Module in Action	4
Scenario 3: Process IDocs Using AIF, Call BAPI/Function Module in Action	6
Scenario 4: Process IDocs with ALE and Write Index Tables with Enabler	8
Related Content.....	10
Copyright.....	11

Introduction

The SAP Application Interface Framework 2.0 provides you with four different scenarios to monitor your IDocs:

- Scenario 1: Monitoring Existing IDocs in Monitoring and Error Handling (p. 3)
- Scenario 2: Process IDoc Using AIF, Call IDoc Function Module in Action (p.4)
- Scenario 3: Process IDocs Using AIF, Call BAPI/Function Module in Action (p. 6)
- Scenario 4: Process IDocs with ALE and Write Index Tables with Enabler (p.8)

This document will give you an overview of the scenarios and provide you with some information about the needed customizing and configuration.

Generate IDoc Structure and Interface Definition

In order to monitor an IDoc with the SAP Application Interface Framework you have to:

- Define an interface and,
- You need to create a structure for an IDoc's basic type.

Those two steps are independent of the scenario. In order to execute those steps a report is provided. Call transaction `/AIF/IDOC_GEN`. On the selection screen you can enter the basic type for which you want to create a structure. Furthermore, you have to enter a prefix. This prefix will be used for the sub-structures and sub-tables that will be created. If you enter multiple basic types you the report will allow you to create an interface for each basic type.

In order to create an interface, enter *Prefix Interface Definition*. This prefix will be used for the interface name. The prefix will be concatenated with a number in order to create unique interfaces within a namespace, in case you have inserted multiple Basic types. You will be able to change the interface name on the next screen. Furthermore, you have to enter a *Variant ID*. Which variant ID you have to choose depends on the IDoc scenario.

Additionally, you can insert the *Package*, where the generated structures should be assigned to as well as a customizing and workbench request. Optionally, it is possible to insert an RFC destination in case the customizing should be created in a different client.

When you execute the transaction you will be able to check the names of the objects that will be created and make changes if necessary. It is possible to change following values:

- Interface Name
- Interface Version
- Raw Data Structure
- Variant ID: the variant ID will define which engines should be maintained for this interface

Note: You can only create one structure for each basic type. However, you can use the generated structure in multiple interfaces. You will be able to update the structure, for example if the components of the basic type have changed.

Scenario 1: Monitoring Existing IDocs in Monitoring and Error Handling

The SAP Application Interface Framework enables you to monitor your existing IDocs in *Monitoring and Error Handling* (transaction /AIF/ERR). The processing of your IDocs will not be influenced by the SAP Application Interface Framework. You will only use the monitoring capabilities of *Monitoring and Error Handling*.

In order to enable your business users to monitor IDocs some configurations have to be made within the SAP Application Interface Framework:

- Create structure from Basic Type
- Define interface in the SAP Application Interface Framework
- Assign message type and basic type to your interface

Use transaction /AIF/IDOC_GEN to create the structure and interface. Enter *Variant ID 01 – Standard IDoc Runtime*. The variant ID will set the corresponding engines for the interface. After you have executed the report the structures and interface should have been created.

Check the generated objects in customizing for the SAP Application Interface Framework (transaction /AIF/CUST). In *Interface Development* → *Define Interfaces*, enter your namespace and select the interface that you just generated. The structure generated with transaction /AIF/IDOC_GEN should be maintained in *SAP Data Structure* and *Raw Data Structure* field.

In *Additional Interface Properties* → *Specify Interface Engines* following engines should have been maintained for your interface:

- Application Engine: IDoc
- Persistence Engine: IDoc
- Selection Engine: IDoc Control Records
- Logging Engine: IDoc Status Records

In customizing activity *Assign IDOC Types* you can define which IDocs should be selected in *Monitoring and Error Handling* for your interface. If you generated the interface with /AIF/IDOC_GEN the basic type should already be maintained. It is recommended that you maintain also the message type. Furthermore, you can maintain *Extension*, *Message Variant* and *Message function* of the IDoc.

Note: If you have already created a structure for a basic type and only want to have a further interface, you can define your interface in *Define Interfaces*, maintain the engines and assign the IDoc types.

In transaction /AIF/ERR you can select the interface and execute the transaction. All IDocs within the selected time range and that have the selected status will be displayed. The last status records of the IDocs can be displayed in *Log Messages* view. You can edit the IDoc's content in *Data Content* view. However, for editing you need the corresponding authorizations and the fields have to be defined as editable in customizing. You can restart and cancel the IDoc.

Scenario 2: Process IDoc Using AIF, Call IDoc Function Module in Action

Data received with an IDoc can be processed with the SAP Application Interface Framework. This enables you to use functionality like checks and value mappings, alerting and the data can be displayed in the *Interface Monitor*. You can call the standard IDoc process function in an action.

In order to process IDoc data in the SAP Application Interface Framework following steps have to be executed:

- Create structure from Basic Type
- Define interface in the SAP Application Interface Framework
- Maintain structure mapping (including checks, field mappings, value mappings etc.)
- Define and assign an action
- Maintain Interface Determination
- (Optional) Maintain recipients for alerting and Interface Monitor
- Maintain IDoc customizing

Use transaction /AIF/IDOC_GEN to generate the structure and the interface. On the selection screen enter *Variant ID 02 – AIF Runtime; Call IDoc function in Action* to set the correct engines for the interface.

After the structure and interface is created check the entries in customizing of the SAP Application Interface Framework (transaction /AIF/ERR). In Interface *Development* → *Define Interface* enter your namespace and select your interface. The structure generated with /AIF/IDOC_GEN should be maintained as *SAP Data Structure* and *Raw Data Structure*. Furthermore, indicator *Move Corresponding Structures* should be set.

In *Additional Interface Properties* → *Specify Interface Engines* following engines should be maintained:

- Application Engine: IDoc
- Persistence Engine: IDoc
- Selection Engine: AIF Index Tables
- Logging Engine: AIF Application Log

If required you can maintain checks, field mappings, value mappings etc. in customizing activity *Define Structure Mappings*.

In *Define Actions* you have to create an action with an action function that will call the standard IDoc process function. Within the action function you can call function module /AIF/IDOC_ACTION_FUNCTION. Pass the name of the standard IDoc process function you would like to call to the function module. Furthermore, you should pass parameter DATA to the function module. The /AIF/IDOC_ACTION_FUNCTION returns a table RETURN_TAB which should be passed to RETURN_TAB of the action function. Below you can see an example of an action function calling the IDoc process function IDOC_INPUT_FLIGHTBOOKING_CREAT.

```

FUNCTION z_action_function .
  "-----
  " "Local Interface:
  " IMPORTING
  " REFERENCE(TESTRUN) TYPE C
  " REFERENCE(SENDING_SYSTEM) TYPE /AIF/AIF_BUSINESS_SYSTEM_KEY
  " OPTIONAL
  " TABLES
  " RETURN_TAB STRUCTURE BAPIRET2
  " CHANGING
  " REFERENCE(DATA)
  " REFERENCE(CURR_LINE)
  " REFERENCE(SUCCESS) TYPE /AIF/SUCCESSFLAG
  " REFERENCE(OLD_MESSAGES) TYPE /AIF/BAL_T_MSG
  "-----

  CALL FUNCTION '/AIF/IDOC_ACTION_FUNCTION'
    EXPORTING
      iv_idoc_function = 'IDOC_INPUT_FLIGHTBOOKING_CREAT'
    TABLES
      return_tab      = return_tab
    CHANGING
      data            = data.

ENDFUNCTION.

```

Note: This function module will only be available if note [1769872](#) is implemented or the first support package for the SAP Application Interface Framework 2.0 is installed in your system. If the note is not installed you have to call following function modules in your action function:

1. /AIF/IDOC_CONVERT_SAP_STRUCT
2. Call IDoc process function
3. /AIF/IDOC_CONVERT_STATREC

Now, you can assign your action to your interface in customizing activity *Define Structure Mapping* → *Assign Actions*.

Furthermore, you have to set up interface determination. In *System Configuration* → *Interface Determination* select *Interface Determination for IDoc Interfaces*. Maintain *Basic Type* and *Message Type* of your interface. In *Assign Interfaces* maintain your interface.

Optionally, you can define fields as editable, assign recipients; define interface specific index tables and selection screens etc. The customizing within the SAP Application Interface Framework is done now. However, you will have to make some changes to the ALE configurations:

1. (Optional) Create Message type in transaction WE81.
2. (Optional) Assign Message type to basic type in transaction WE82.
3. Call transaction BD51 and check if function module /AIF/IDOC_INBOUND_PROCESS_FUNC is already maintained (Input type: 1 and Dialog allowed: blank). /AIF/IDOC_INBOUND_PROCESS_FUNC is a generic function module delivered with the SAP Application Interface Framework that will trigger processing of the IDoc data in the SAP Application Interface Framework.
4. Call transaction WE57 to assign the function module to logical message and IDoc type. Maintain following entries:
 - Function module: /AIF/IDOC_INBOUND_PROCESS_FUNC
 - Function type: Function module
 - Basic type: the basic type of your interface
 - Message type: the message type of your interface
 - Direction: Inbound
5. Maintain an inbound process code in transaction WE42 and select Processing by function module in Processing type. Save the entry and maintain function module /AIF/IDOC_INBOUND_PROCESS_FUNC.
6. Maintain partner profile in transaction WE20.

Note: Depending on your concrete scenario the steps that you have to perform might differ from the ones mentioned above. For example, instead of creating a new process code it might be enough to change an existing process code to call function module /AIF/IDOC_INBOUND_PROCESS_FUNC.

To test your implementation of the IDoc interface you can use transaction WE19. Select your interface in *Monitoring and Error Handling* (/AIF/ERR). You will see the processed IDocs, the log messages and the content. If you have the necessary authorization and fields are defined as editable you can change the field content of the erroneous IDocs. You can restart or cancel the IDocs. In case you have assigned the interface to a recipient, you are assigned to, you should also see the IDocs in the *Interface Monitor* (/AIF/IFMON).

Scenario 3: Process IDocs Using AIF, Call BAPI/Function Module in Action

The SAP Application Interface Framework allows you to process data received by an IDoc. You can map the data to any other internal structure, for example you want to process the data with a BAPI which is called in an action.

In order to be able to call a BAPI or any other function in an action function you have to perform following steps:

- Select BAPIs, (custom) functions you want to call in the action function
- Create an SAP structure that fits to the parameters required by the BAPIs and functions
- Create structure from Basic Type
- Define interface in the SAP Application Interface Framework
- Maintain structure mapping (including checks, field mappings, value mappings etc.)
- Define and assign an action
- Maintain Interface Determination
- (Optional) Maintain recipients for alerting and Interface Monitor
- Maintain IDoc customizing

Create a structure that fits to the functions you would like to call in the interface's actions. This structure will be your interface's SAP Data structure.

Use transaction /AIF/IDOC_GEN to generate a structure for your basic type and an interface. On the selection screen enter *Variant ID 03 – AIF Runtime; User Specific Action*.

After you have executed the report go to customizing of the SAP Application Interface Framework and select *Interface Development → Define Interface*. Select the interface you created. *The Raw Data Structure* field should be filled with the structure you generated with the report. The *SAP Data Structure* field should be empty. Enter the name of the structure you created based on the functions you want to call in the action.

In *Additional Interface Properties → Specify Interface Engines* following engines should be maintained:

- Application Engine: IDoc
- Persistence Engine: IDoc
- Selection Engine: AIF Index Tables
- Logging Engine: AIF Application Log

Since the raw data structure of your interface will differ from its SAP data structure you have to maintain structure and field mappings in *Define Structure Mappings*. Furthermore, you can assign checks and value mappings, etc. You have to create an action that will call the BAPI or function in an action function. To call the action the data mapped to the SAP data structure will be used.

Additionally, you can define key fields, interface specific index tables and an interface specific selection screen, set fields as editable. To display the IDocs received with this interface and to use alerting you can assign a recipient to this interface.

Furthermore, you should set up interface determination in *System Configuration → Interface Determination → Interface Determination for IDoc Interfaces*. Maintain/select the basic type and message type of your interface, and assign your interface.

Then you will have to set up to some configurations in for the ALE runtime.

1. (Optional) Create message type in transaction WE81.
2. (Optional) Assign message type to basic type in transaction WE82.
3. Call transaction BD51 and check if function module /AIF/IDOC_INBOUND_PROCESS_FUNC is already maintained (Input type: 1 and Dialog allowed: blank). /AIF/IDOC_INBOUND_PROCESS_FUNC is a generic function module delivered with the SAP Application Interface Framework that will trigger processing of the IDoc data in the SAP Application Interface Framework.
4. Call transaction WE57 to assign the function module to logical message and IDoc type.
5. Maintain an inbound process code in transaction WE42 and select *Processing by function module* in *Processing type*. Save the entry and maintain function module /AIF/IDOC_INBOUND_PROCESS_FUNC.
6. Maintain partner profile in transaction WE20

To test your implementation of the interface you can use transaction WE19.

Call the *Monitoring and Error Handling (/AIF/ERR)* transaction and select your interface. The IDocs processed with this interface fitting to the selection criteria should be displayed. You can edit, restart and cancel the IDocs, if you have the corresponding authorizations. In case you maintained a recipient the IDocs should also be displayed in the *Interface Monitor (/AIF/IFMON)*.

Scenario 4: Process IDocs with ALE and Write Index Tables with Enabler

This scenario is applicable, if:

- You would like to monitor an existing IDoc type with the SAP Application Interface Framework
- You do not want to process the data with the SAP Application Interface Framework, but
- You want to use functionality like alerting, interface specific index tables and interface specific selection screens

The enabler for IDocs is a class that you can use to perform some SAP Application Interface Framework specific actions. To use the enabler you have to create your own custom specific process function. Within this process function you have to call the standard IDoc process function. After calling the IDoc process function you call static method TRANSFER_TO_AIF of class /AIF/CL_ENABLER_IDOC. Pass the control records and the status records to this method. TRANSFER_TO_AIF will then create index table entries and alerts.

```
DATA: lt_idoc_status TYPE bdtidocsta,
      ls_idoc_control TYPE edidc.
```

```
* 1. Call standard IDoc process Function
CALL FUNCTION <name of IDoc process function>
  EXPORTING
    input_method          = input_method
    mass_processing       = mass_processing
  IMPORTING
    workflow_result      = workflow_result
    application_variable = application_variable
    in_update_task       = in_update_task
    call_transaction_done = call_transaction_done
  TABLES
    idoc_contr1          = idoc_contr1
    idoc_data            = idoc_data
    idoc_status          = idoc_status
    return_variables     = return_variables
    serialization_info   = serialization_info
  EXCEPTIONS
    wrong_function_called = 1
    OTHERS                = 2.
IF sy-subrc <> 0.
* Implement suitable error handling here
ENDIF.

lt_idoc_status = idoc_status[].
READ TABLE idoc_contr1 INTO ls_idoc_control INDEX 1.

* 2. Call AIF Enabler in order to create AIF index table entries, raise alerts etc.
CALL METHOD /aif/cl_enabler_idoc=>transfer_to_aif
  EXPORTING
    is_idoc_control_rec = ls_idoc_control
    it_bdidocstat       = lt_idoc_status.
```

Use transaction /AIF/IDOC_GEN to generate a structure for your basic type and an interface. On the selection screen enter *Variant ID 05 – AIF Enabler; no Application Log*.

After you have executed the report go to customizing of the SAP Application Interface Framework and select *Interface Development*→*Define Interface*. Select the interface you created. *The Raw Data Structure* field and the *SAP Data Structure* field should be filled with the structure you generated with the report.

In *Additional Interface Properties*→*Specify Interface Engines* following engines should be maintained:

- Application Engine: IDoc
- Persistence Engine: IDoc
- Selection Engine: AIF Index Tables

- Logging Engine: IDoc Status Records

Furthermore, you should set up interface determination in *System Configuration* → *Interface Determination* → *Interface Determination for IDoc Interfaces*. Maintain/select the basic type and message type of your interface, and assign your interface.

Furthermore, you might have to go through following steps:

- (optional) Create message type (transaction WE81)
- (optional) Assign message type to basic type (transaction WE82)
- Maintain characteristics of your custom inbound process function module (transaction BD51; Input type: 1 and Dialog allowed: blank)
- Assign function module to message type and basic type (transaction WE57)
- Create inbound process code (transaction WE42). Select *Processing by function module* in *Processing type*. Save the entry and maintain your custom function module
- Maintain partner profile (transaction WE20)

To test your implementation you can use transaction WE19.

Call the *Monitoring and Error Handling (/AIF/ERR)* and select your interface. The IDocs processed with this interface fitting to the selection criteria should be displayed. You can edit, restart and cancel the IDocs, if you have the corresponding authorizations. In case you maintained a recipient the IDocs should also be displayed in the *Interface Monitor (/AIF/IFMON)*.

Related Content

[Cookbook for the SAP Application Interface Framework](#)

[IDoc Scenarios](#)

Copyright

© Copyright 2012 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Oracle Corporation.

JavaScript is a registered trademark of Oracle Corporation, used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.