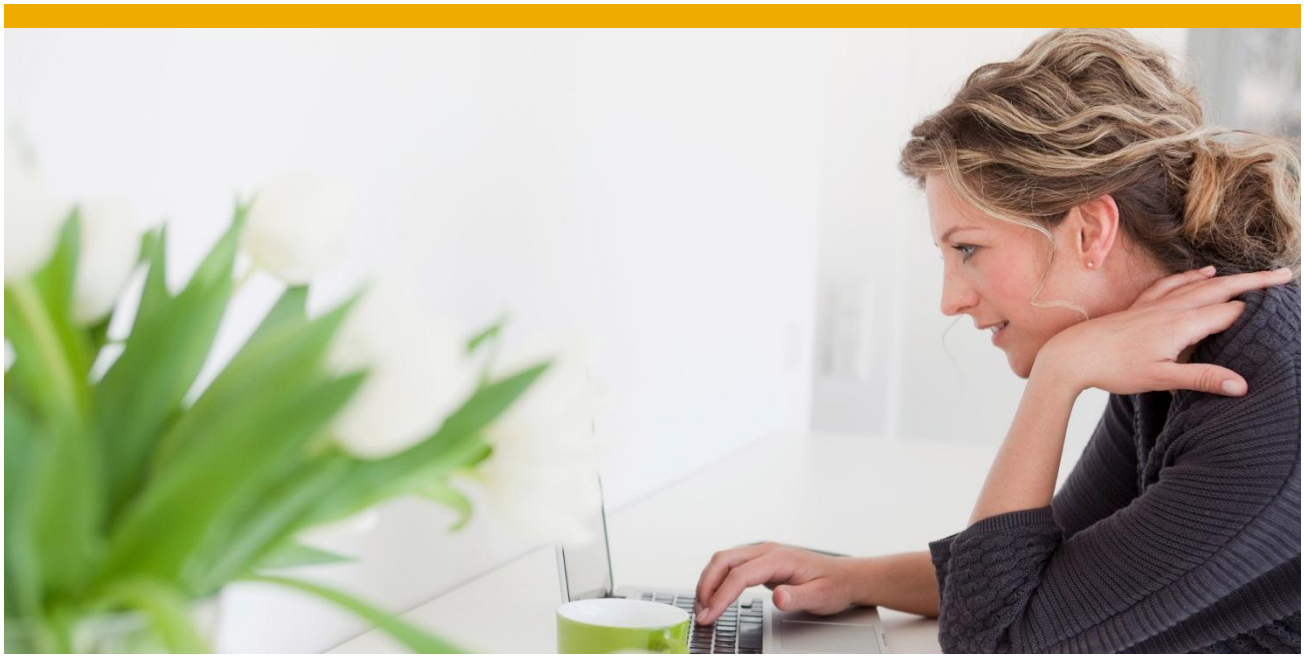

Outbound Proxy Interfaces with Persistence Layer

Store Source Data of an Outbound Proxy Interface in the Persistence Layer of the SAP Application Interface Framework

TABLE OF CONTENTS

SPECIAL REMARKS ABOUT OUTBOUND INTERFACES 3
OUTBOUND PROXY INTERFACE WITH PERSISTENCE OF SOURCE DATA 4
EXAMPLE OF AN AIF OUTBOUND PROXY INTERFACE 5
Perquisites 5
Service Interface 5
Create an SAP Structure 5
Define AIF Interface and Structure Mapping 6
Define Interface 6
Specify Interface Engines 6
Create Structure Mapping 6
Report to Trigger Sending of Data 8

In *Monitoring and Error Handling* of the SAP Application Interface Framework the source data can be displayed. However, in case of outbound proxy messages the source data is usually not stored anywhere. Therefore, it is not possible to display data content of an outbound proxy message. An own persistence is delivered with the SAP Application Interface Framework 2.0. You can use this persistence layer to store the source data of an outbound interface. If the source data is persisted you are not only able to display it in *Monitoring and Error Handling*, but can also correct errors and restart or cancel a message.

SPECIAL REMARKS ABOUT OUTBOUND INTERFACES

In contrast to inbound interfaces, where processing in the SAP Application Interface Framework is triggered by the proxy class implementation, outbound interfaces have to be manually invoked. This means that a report, transaction, user exit implementation, and so on needs to actively call the function module /AIF/SEND_WITH_PROXY. When calling the method, at least the interface keys (namespace, interface name, and interface version) and the SAP data structure need to be provided. The following figure illustrates outbound message processing with the SAP Application Interface Framework (data flow is from left to right):

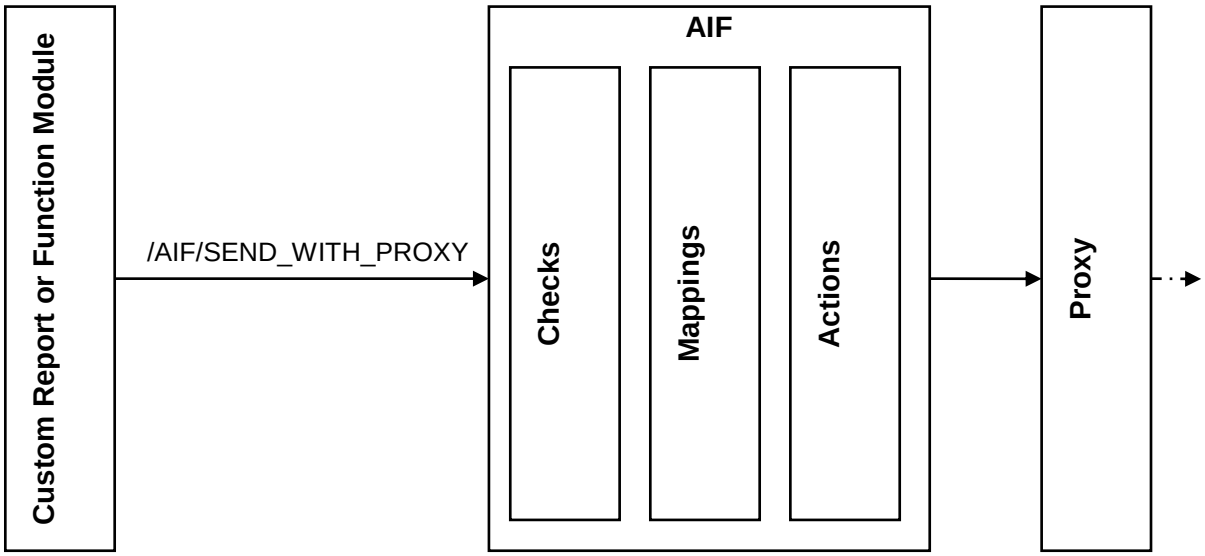

Figure 1: Outbound Message Processing

Outbound message processing with the SAP Application Interface Framework differs from inbound processing in the following aspects:

	Inbound	Outbound
Terminology	Raw data structure is source structure SAP data structure is target structure	SAP data structure is source structure Raw data structure is target structure
Invocation	Is invoked in proxy class implementation by calling /aif/cl_enabler_proxy=>process_message	Is invoked in custom implementation by calling /AIF/SEND_WITH_PROXY
Population of source fields	Raw data structure that is passed to SAP Application Interface Framework was populated by local Integration Engine	SAP data structure that is passed to SAP Application Interface Framework is populated before calling SAP Application Interface Framework manually (for example, in a report or function module)

Data handling in SAP Application Interface Framework	Raw data is received, mappings are performed, then actions are executed	SAP data structure data is received, mappings are performed, (actions are executed), raw data is sent out via proxy
Action processing	You always have actions to do some processing in the system	You usually do not have actions, since you simply want to send data and not process anything in the system

Note: The data is only sent via the outbound proxy after a COMMIT WORK is executed. You can enforce a COMMIT WORK by setting the option Separate Commit in the interface definition, by setting the indicator DO_COMMIT of /AIF/SEND_WITH_PROXY to 'X' (this is the default value), or by submitting a COMMIT WORK manually in the application after calling /AIF/SEND_WITH_PROXY.

OUTBOUND PROXY INTERFACE WITH PERSISTENCE OF SOURCE DATA

The SAP Application Interface Framework enables you to monitor proxy outbound interfaces as well. You can map the data from the SAP data structure to the raw data structure of the interface and add checks and value mappings etc. However, in case of an outbound proxy message the source data is not persisted and therefore it cannot be displayed in *Monitoring and Error Handling*. Since the source data is not stored restarting an erroneous outbound message is not possible. In order to solve this you can store the data in the persistence of the SAP Application Interface Framework.

This document explains how such an interface can be developed. First of all you have to define an interface in the Enterprise Service Repository. Afterwards you can generate an ABAP proxy in your backend system.

In customizing activity *Define Interfaces* of the SAP Application Interface Framework you have to define an interface. Maintain an SAP data structure. How this structure will look like depends on the data provided by your report or function module, that will trigger the sending of the message. Enter your outbound proxy class into *Proxy Class Outbound* field. After pressing enter the raw data structure and record type in raw structure should be filled automatically.

Afterwards you have to specify the interface's engines. Maintain application engine XML and persistence engine XML, since the interface will persist the data in the Application Interface Frameworks persistence layer.

Go to customizing activity *Define Structure Mapping* and create structure and field mappings and assign checks and value mappings etc.

Furthermore, you need a report or function module that will trigger the sending of the data. Within the report or function module you have to call function module /AIF/SEND_WITH_PROXY. You have to pass the namespace, interface name and version to the function module. Additionally, you have to pass the data that the interface should process to the function module. The data should have the format of your interface's SAP structure.

```
CALL FUNCTION '/AIF/SEND_WITH_PROXY'
  EXPORTING
    ns                          = 'X102_0'
    ifname                      = 'OUT_PERS'
    ifversion                   = '1'
 CHANGING
   sap_struct                  = ls_sap_structure
 EXCEPTIONS
   persistency_error           = 1
   status_update_failed        = 2
   OTHERS                      = 3
          .
IF sy-subrc <> 0.
```

```
* Implement suitable error handling here
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
  WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.
```

If you send some data with your interface, you should be able to see the messages in transaction /AIF/ERR. Since the data was persisted on the persistence layer delivered with the SAP Application Interface Framework you will be able to display a message's content in Data Content view. If you have defined fields as changeable and if you have the corresponding authorization you can change a field's content. Furthermore, you can restart or cancel a message.

EXAMPLE OF AN AIF OUTBOUND PROXY INTERFACE

Perquisites

Service Interface

Before you can start to create your AIF interface you have to create your outbound proxy. Therefore, you have to perform following steps:

- Define service interface in the Enterprise Service Repository
- Generate outbound proxy in transaction SPROXY

The figure below shows the message Message Type of the Service interface used in the example:

Name	Category	Type	Occurrence
ZGTP_TEST_RAW_MT	Element	ZGTP_TEST_RAW	
RAW_GLOBAL	Element	ZGTP_TEST_RAW_GLOBAL	0..1
NS	Element	xsd:string	1
IFNAME	Element	xsd:string	1
GLOBAL_C1	Element	xsd:string	0..1
GLOBAL_C2	Element	xsd:string	0..1
RAW_HEADER_AND_ITEM	Element	ZGTP_TEST_RAW_HEADER_AND_ITEM	0..unbounded
RAW_HEADER_T	Element	ZGTP_TEST_RAW_HEAD	0..unbounded
RAW_ITEM_T	Element	ZGTP_TEST_RAW_ITEM	0..unbounded
RAW_ADD_DATA_T	Element	ZGTP_TEST_RAW_ADD_DATA	0..unbounded
ADD_DATA_KEY	Element	xsd:string	0..1
ADD_DATA1	Element	xsd:string	0..1
ADD_DATA2	Element	xsd:string	0..1
ADD_DATA3	Element	xsd:string	0..1
ADD_DATA4	Element	xsd:string	0..1
ADD_DATA5	Element	xsd:string	0..1
ADD_DATA6	Element	xsd:string	0..1

Create an SAP Structure

Since this is an outbound interface the SAP structure is the source structure. This structure contains the data that you will enter in your report. The SAP structure for this example looks as follows:

SAP Structure (ZAIF_X102_SAP_FLIGHT_BOOKING):

Component	Component Type
CUSTOMER_DATA	ZAIF_X102_CUSTOMER_DATA

FLIGHT_BOOKINGS	ZAIF_X102_FLIGHT_BOOKING_TAB

Sub-Structure CUSTOMER_DATA (ZAIF_X102_CUSTOMER_DATA)

Component	Component Type
CUSTOMERNUMBER	S_CUSTOMER
CUST_TYPE	S_CUSTTYPE

Sub-Table FLIGHT_BOOKINGS (ZAIF_X102_FLIGHT_BOOKING_TAB) has the line type ZAIF_X102_FLIGHT_BOOKING_2 which has following components

Component	Component Type
AIRPORT_FROM	S_FROMAIRP
AIRPORT_TO	S_TOAIRP
AIRLINEID	S_CARR_ID
FLIGHT_DATE	S_DATE
CLASS	S_CLASS
AGENCY	S_AGNCYNUM

Define AIF Interface and Structure Mapping

Define Interface

Go to customizing of the SAP Application Interface Framework (transaction /AIF/CUST) and go to *Define Interface*. Select a namespace and create a new interface, for example X102_0/OUT_FLBOOK/1.

Enter your SAP data structure (e.g. ZAIF_X102_SAP_FLIGHT_BOOKING). Furthermore, you have to enter your outbound proxy class (e.g. ZAIF_X102CO_TEST_OUTBOUND06). After you press enter the raw data structure and the record type should be filled automatically.

Specify Interface Engines

Since the data of the SAP data structure will be persisted on the AIF's own persistence you have to set the Interface Engines accordingly. Use following engines:

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

Create Structure Mapping

Go to customizing activity *Define Structure Mapping*. Create a structure mapping from CUSTOMER_DATA to RAW_GLOBAL. Therefore, enter CUSTOMER_DATA into Source Structure and go to Assign Destination Structure. Enter Number of Structure Mapping, e.g 10. Enter RAW_GLOBAL into Destination Structure.

Go to Define Field Mapping to create the field mappings displayed in the table below.

Field in Destination Structure	Fieldname 1	Namespace	Value Mapping
GLOBAL_C1	CUSTOMERNUMBER		

GLOBAL_C2	CUSTOMERNUMBER	X102_0	GET_CUSTOMER_NAME

The customer name will be passed to field GLOBAL_C2. The customer name is derived from field CUSTOMERNUMBER. Therefore, a Value Mapping GET_CUSTOMER_NAME to retrieve the customer's name from the database needs to be created. Enter the value mapping name and confirm that the new value mapping should be created. Afterwards, double click on your newly created value mapping to define your value mapping in the following way:

Field	Value
Value Mapping Description	Select customer name
Table Name	SCUSTOM
Field Name	NAME
Where Condition for Select Statement	ID = '$1'
Single or Multiple Value Mapping	None
Customizing or Master Data	None

Optional: Assign a descriptive error message to support your business user in solving the error. Save the value mapping.

Go back to customizing activity *Define Structure Mapping* and go to Select Source Structure. Enter source structure FLIGHT_BOOKINGS. Go to Assign Destination Structure and enter destination structure RAW_ADD_DATA_T.

In Define Field Mappings create following simple 1:1 mappings:

Field in Destination Structure	Fieldname 1
ADD_DATA_KEY	AGENCY
ADD_DATA1	AIRPORT_FROM
ADD_DATA2	AIRPORT_TO
ADD_DATA3	AIRLINEID
ADD_DATA4	FLIGHT_DATE
ADD_DATA5	CLASS

Furthermore, the Connection ID is needed. The connection ID can be derived from AIRPORT_FROM, AIRPORT_TO, FLIGHT_DATE and AIRLINEID. Enter those fields into Fieldname 1 to Fieldname 4. Enter Value Mapping e.g. X102_0, VM_CONNECTID. Press enter and confirm that a new value mapping should be created. Double click on the value mapping and create the value mapping as follows:

Field	Value
Value Mapping Description	Determines connection by airports/date/airline
Table Name	SFLIGHTS
Field Name	NAME
Where Condition for Select Statement	AIRPFROM = '$1' AND AIRPTO = '$2' AND FLDATE = '$3' AND CARRID = '$4'

Single or Multiple Value Mapping	None
Customizing or Master Data	None

Optional: Assign a descriptive error message to support your business user in solving an error.

Save the value mapping and return to your structure mapping.

Save your structure mapping.

Report to Trigger Sending of Data

In order to test your interface create a small report where you can enter some data to create a flight booking. Move the data to your SAP Structure. Call function module /AIF/SEND_WITH_PROXY and pass your SAP structure to the function modules changing parameter SAP_STRUCT. Furthermore, you have to pass the interface keys.

```
REPORT  zaif_test_x102_send_flbookings.

PARAMETERS: p_agency  TYPE s_agncynum DEFAULT 120,
            p_custno  TYPE s_customer,
            p_airfro  TYPE s_fromairp,
            p_airto   TYPE s_toairp,
            p_carrid  TYPE s_carr_id,
            p_fldate  TYPE s_date,
            p_class   TYPE s_class.

DATA: ls_sap_structure TYPE zaif_x102_sap_flight_booking,
      ls_flbooking     TYPE zaif_x102_flight_booking_2.

ls_sap_structure-customer_data-customernumber = p_custno.

ls_flbooking-agency = p_agency.
ls_flbooking-airport_from = p_airfro.
ls_flbooking-airport_to = p_airto.
ls_flbooking-airlineid = p_carrid.
ls_flbooking-flight_date = p_fldate.
APPEND ls_flbooking TO ls_sap_structure-flight_bookings.

CALL FUNCTION '/AIF/SEND_WITH_PROXY'
  EXPORTING
    ns                  = 'X102_0'
    ifname              = 'OUT_FLBOOK'
    ifversion           = '1'
  CHANGING
    sap_struct          = ls_sap_structure
  EXCEPTIONS
    persistency_error   = 1
    status_update_failed = 2
    missing_keys        = 3
    OTHERS              = 4.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
    WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ELSE.
  WRITE: 'Flight Booking transfered to AIF'.
ENDIF.
```

Activate the report and execute it. Enter some data in to the selection screen. Afterwards you can check the *Monitoring and Error Handling* transaction. Make sure to select all status. If your data was ok the data should

have been sent via your outbound proxy. If you had an error, for example because you entered a flight that does not exist, the message was not sent. Since the data you entered into the report is persisted in the AIF's persistence you are now able to have a look at the data. If you have defined some changeable fields you might also be able to correct the error. You can then restart the message and the data should be sent via your outbound proxy.

www.sap.com