Send, Receive, and Process Interactive Forms via Email in AS ABAP.

Jeff Gebo

NetWeaver RIG, SAP Labs LLC

Introduction

Creating Interactive Forms in AS ABAP

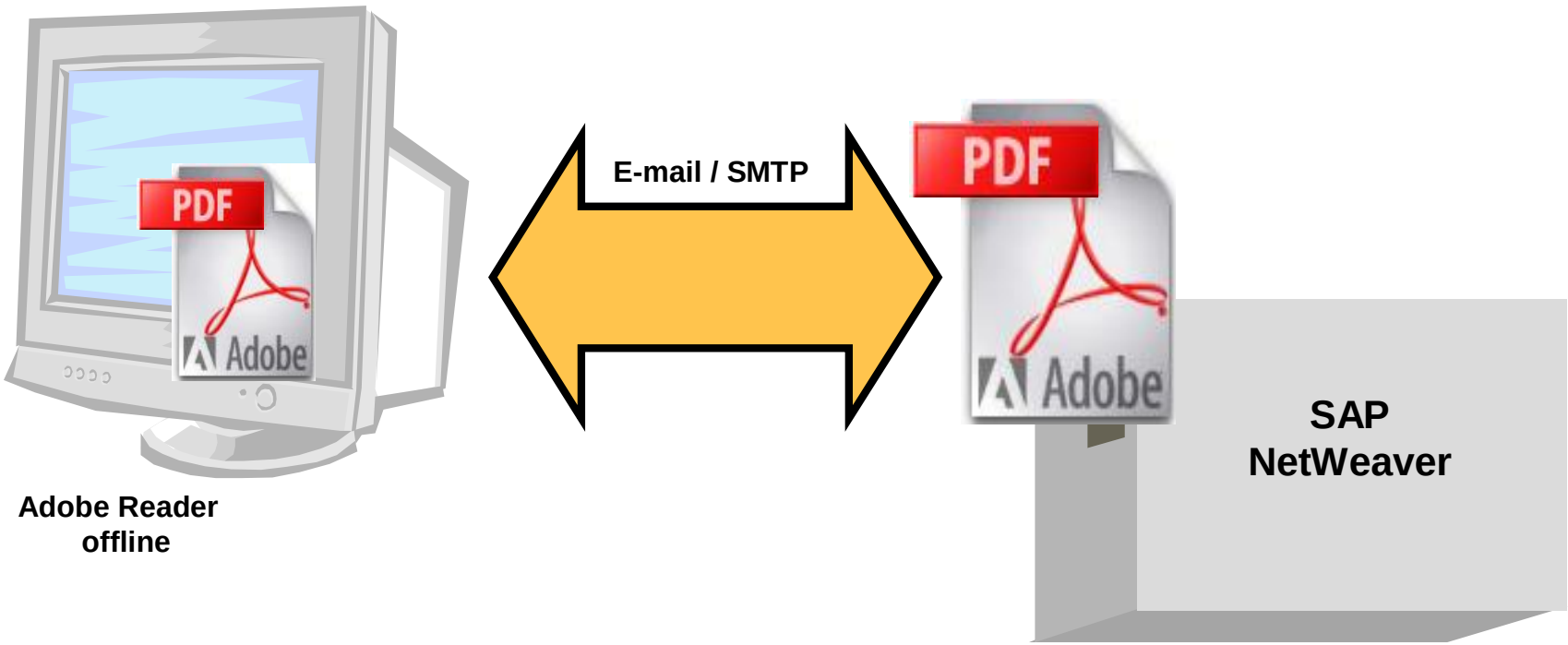
SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Interactive Form Offline Scenario



- **User fills out form offline, submits form via email when online.**
- **No direct SAP access needed.**
  - **Client email program needed to send and receive emails with attachments.**

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Creating Interface & Form

To create an Interactive Form you must create the form interface and then the form itself in transaction SFP.

Form Interface

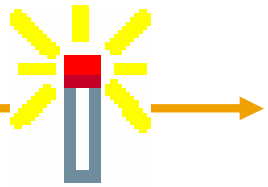
- **Represents the data structures you wish to display or have the user enter on the Interactive Form.**
- **Can be simple types, structures or tables.**

Form

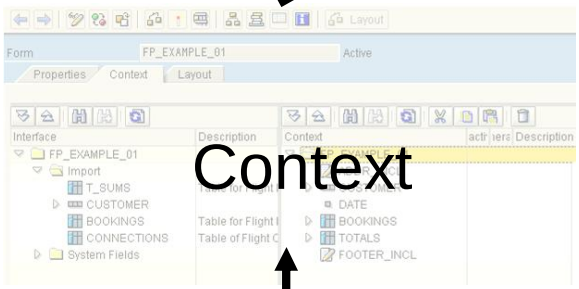
- **Create a form and assign the form interface to it. Data elements in the form interface can then be used in the form.**
- **LiveCycle Designer is displayed in the “Layout” mode of the form.**

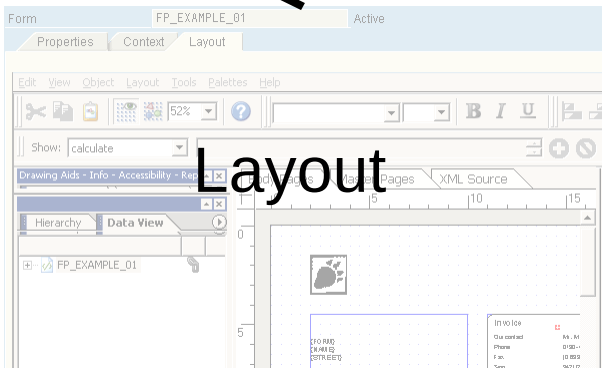
The Tools Involved (Design Time)

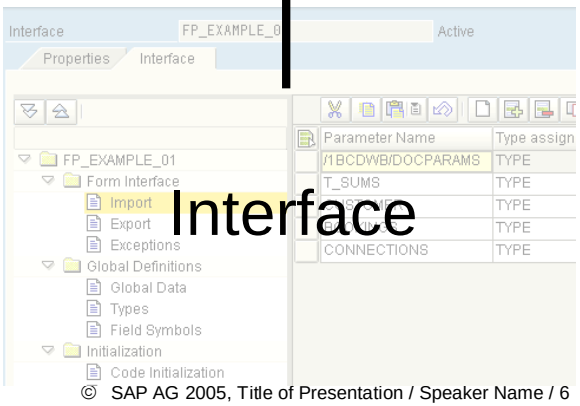
Form Template



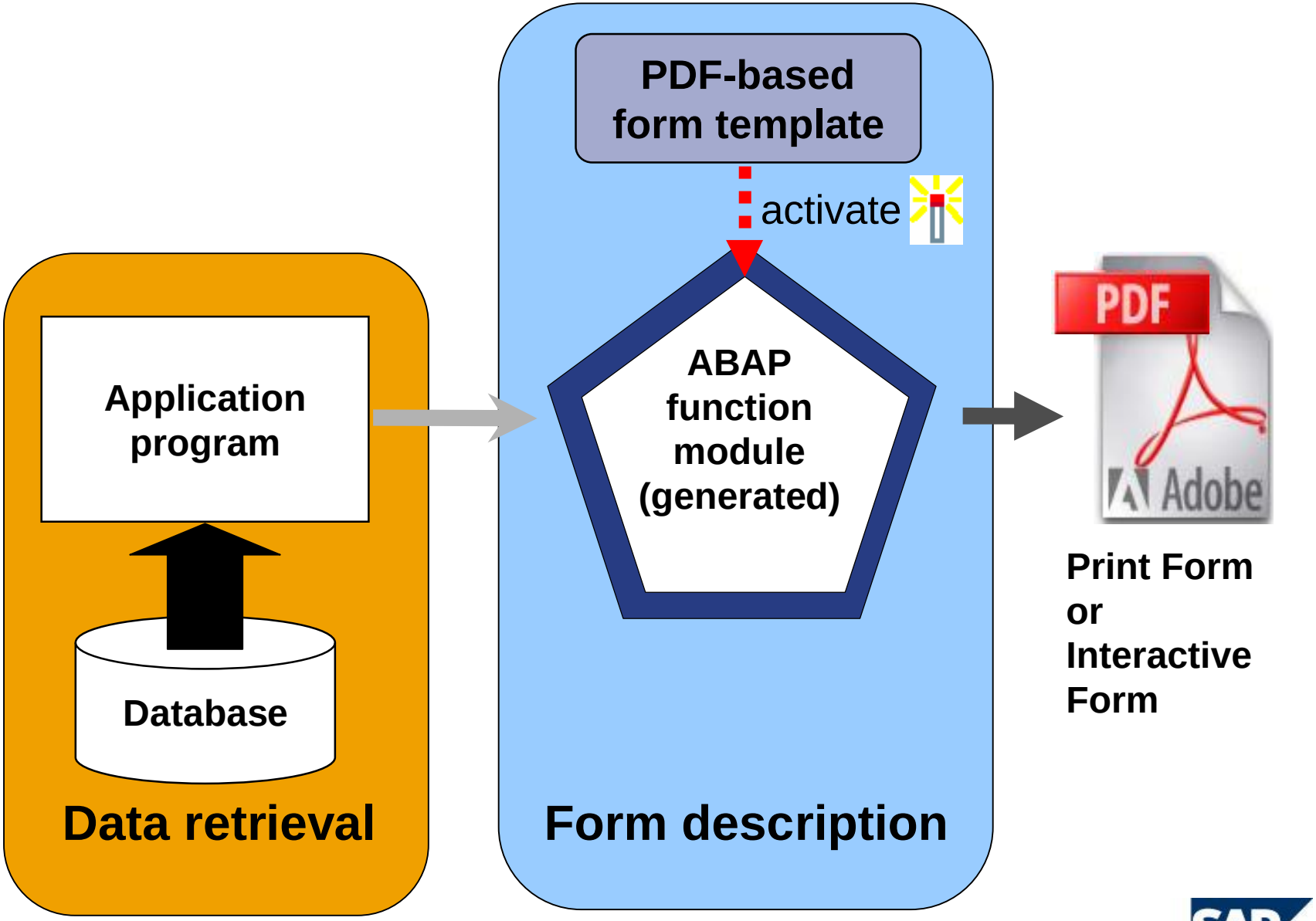
```
FUNCTION
/1BCDWB/SM00000001.
DATA: %OUTPAR TYPE
SFPOUTPAR,
%DOCPAR TYPE
SFPDOCPAR,
...
```



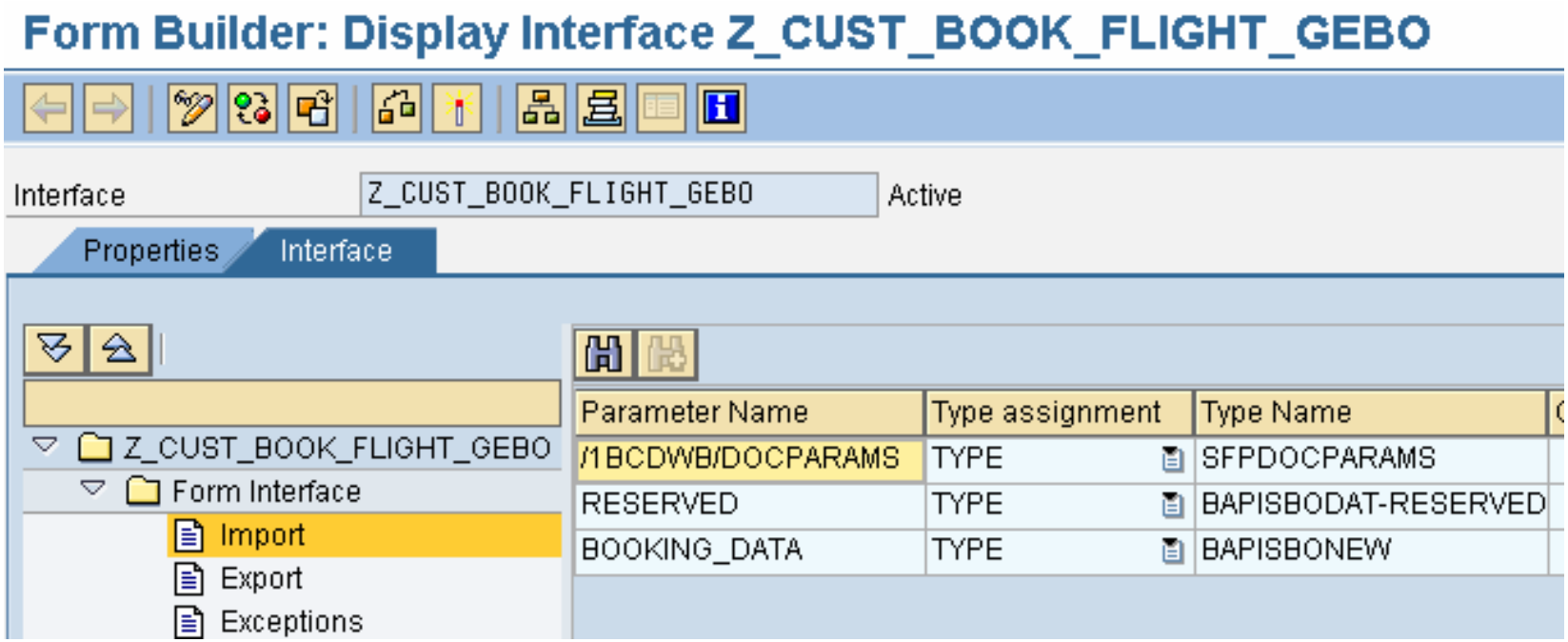




What Happens at Run Time

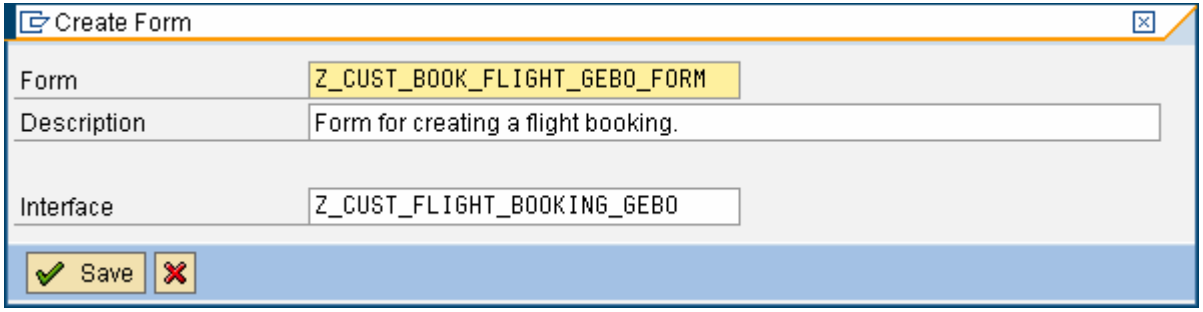


Form Interface Example

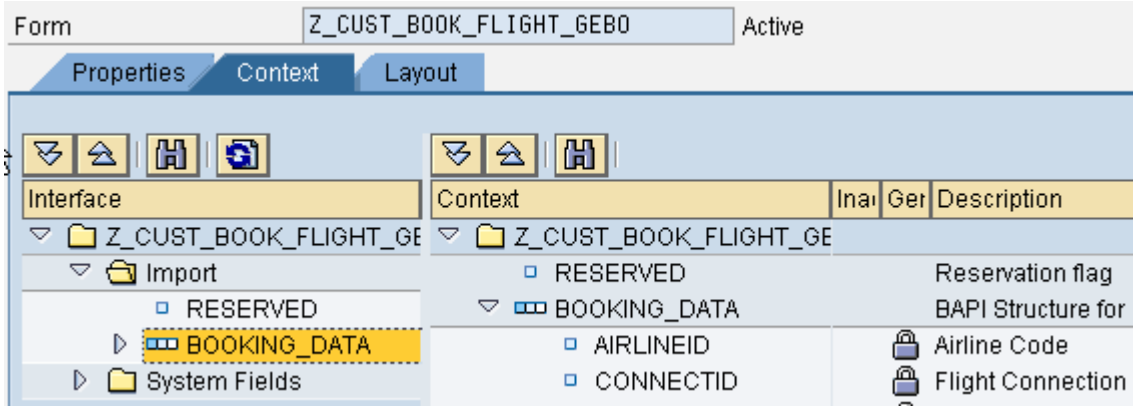


- **Use transaction SFP to create/modify/view Form Interfaces.**
- **Add your data elements to the Import parameter list.**

Creating Forms



- **When creating a form you must assign it an interface.**

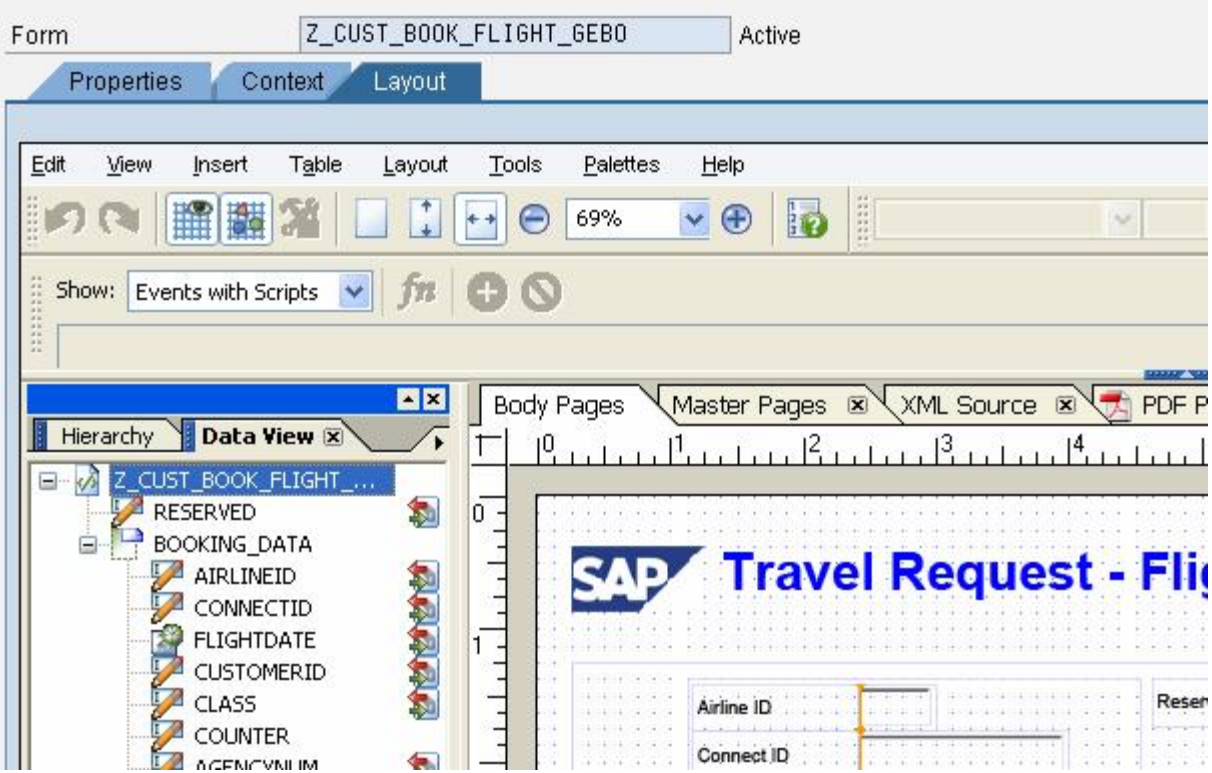


- **Create the forms context by dragging and dropping the interface's import elements to the forms context.**

- **These context elements can then be bound to input fields on the form.**

Designing Forms

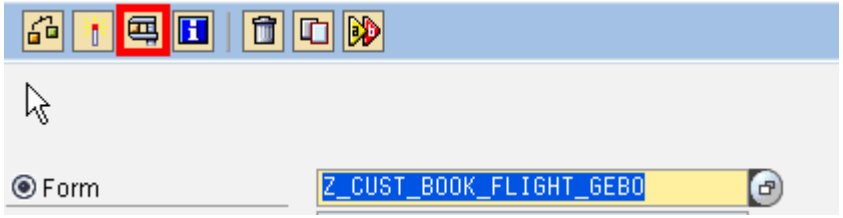
Select the Layout tab of the Form Builder, this will start the LiveCycle Designer.



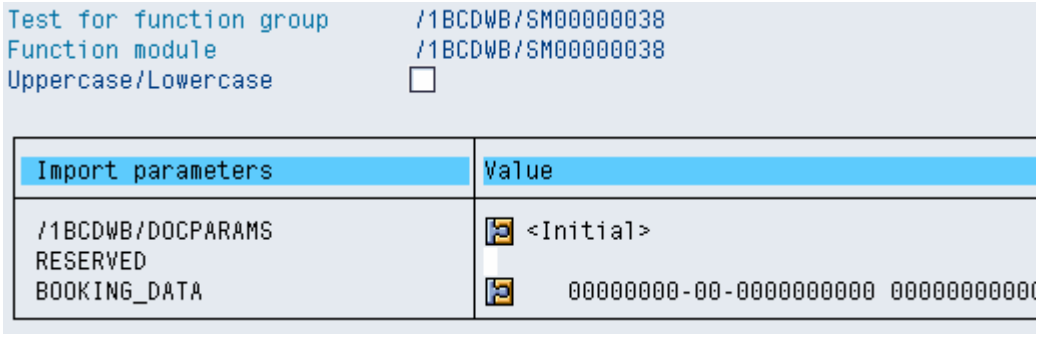
Design your form as you would a standard print form. Just make sure not to set the input fields to read only!

Testing Forms

You can test your form by clicking the test icon.



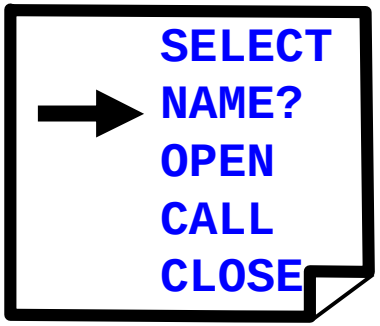
This will display a function module that contains the same input parameters as the form's interface – just enter data into the data elements.



Sections of the Application Program

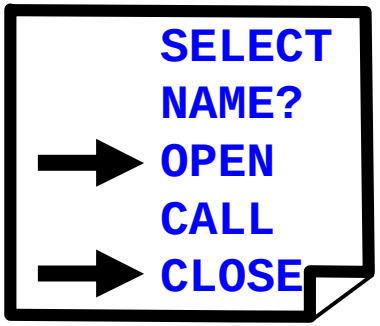
```
* (1) Data retrieval and processing
SELECT ... FROM ...
...

* (2) Find out name of generated function module

CALL FUNCTION 'FP_FUNCTION_MODULE_NAME'...

* (3) Start form processing

CALL FUNCTION 'FP_JOB_OPEN'...

LOOP AT ...
* (4) Call function module dynamically
   CALL FUNCTION <generated function module> ...
ENDLOOP.

* (5) End form processing
CALL FUNCTION 'FP_JOB_CLOSE'...
```

FP_FUNCTION_MODULE_NAME



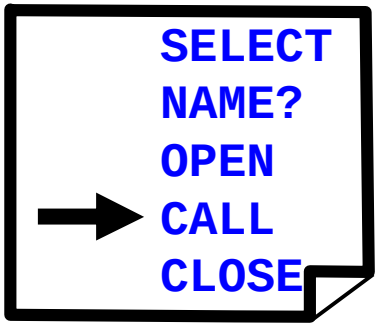

```
DATA:
   form                TYPE fpwbformname,
   fm_name             TYPE funcname.

* (2) Find out name of generated function module
CALL FUNCTION 'FP_FUNCTION_MODULE_NAME'
   EXPORTING i_name    = form
   IMPORTING
      e_funcname       = fm_name.
```

Starting and Ending Form Processing




```
DATA:
   fp_outputparams   TYPE sfpoutputparams
   fp_outputparams-getpdf = 'X'.
...

* (3) Start form processing
CALL FUNCTION 'FP_JOB_OPEN'...
* set output parameters like printer, preview...
   CHANGING ie_outputparams = fp_outputparams ...


...

* (5) End form processing
CALL FUNCTION 'FP_JOB_CLOSE'...
```

Calling the Generated Function Module




```
DATA:
   fp_docparams   TYPE sfpdocparams,
   fm_name        TYPE funcname,
   fp_result      TYPE TYPE fpformoutput, ...

   fp_docparams-fillable = 'X'.
   fp_docparams-langu = customer_language.

* (4) Call function module dynamically
    CALL FUNCTION fm_name
       EXPORTING
         /1bcdwb/docparams  = fp_docparams
         bookings  = it_bookings
       IMPORTING
       /1bcdwb/formoutput = fp_result
       EXCEPTIONS
         OTHERS             = 1.

   data: pdf type fpformoutput-pdf.
   pdf = fp_result-pdf.
```

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

SMTP Plug-in

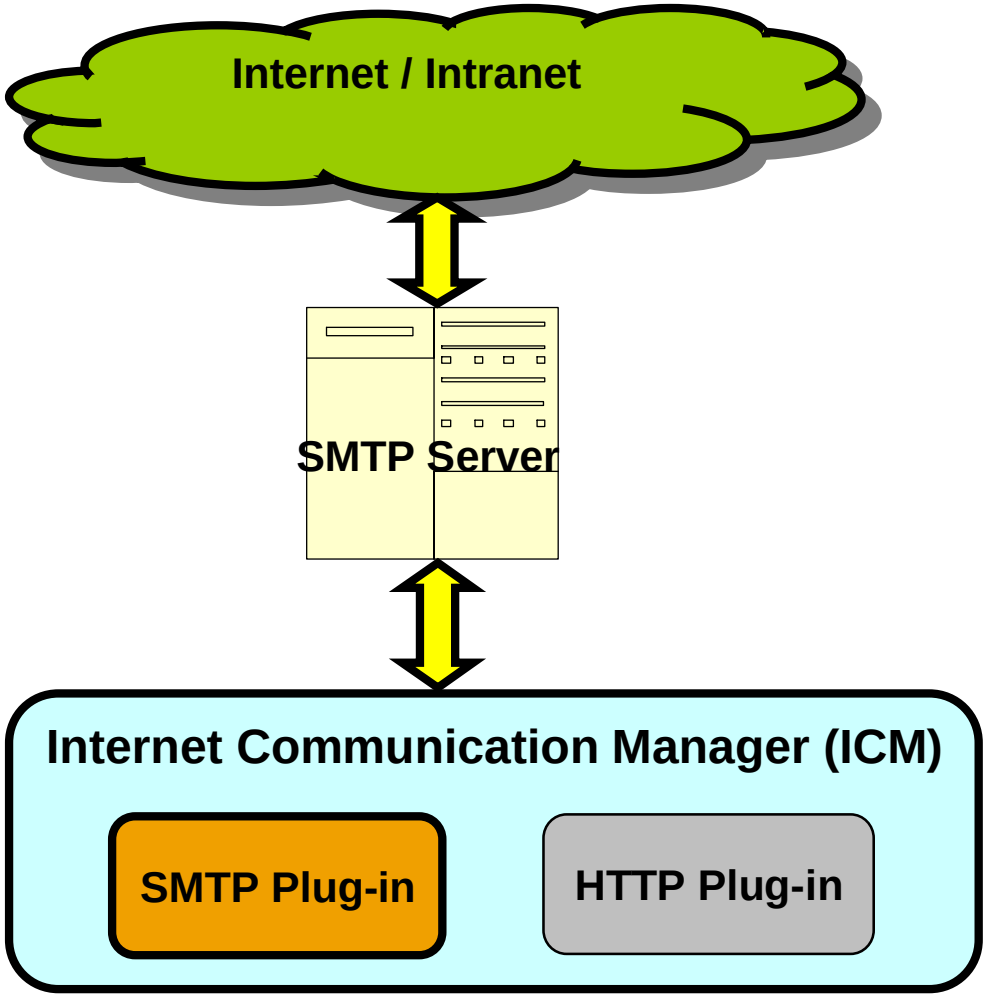
ICM handles communication protocols via Plug-Ins.

- **Currently there are plug-ins for SMTP and HTTP. This can be easily extended in the future for new protocols.**

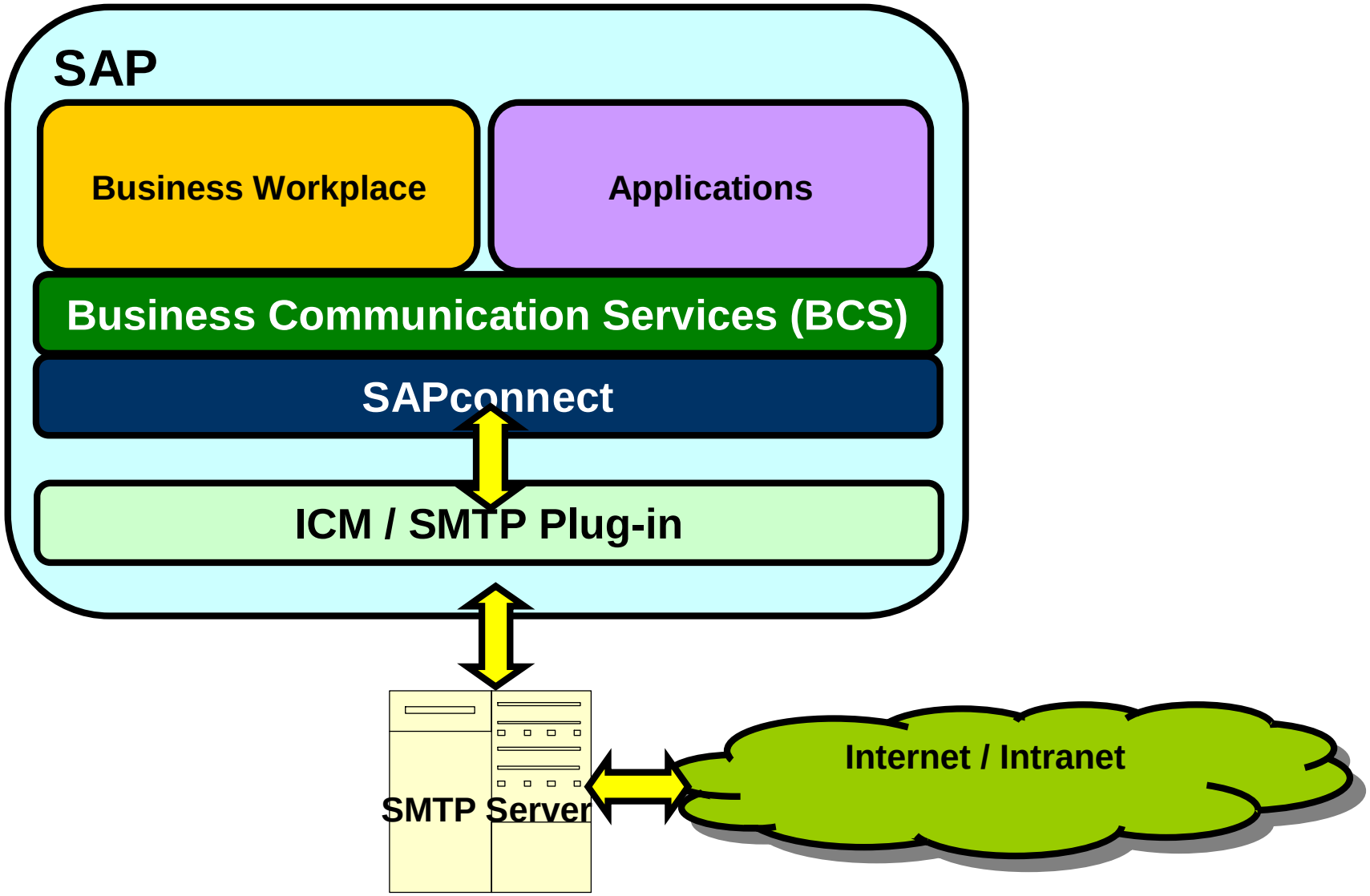
- **Configure SMTP plug-in with profile parameters:**

icm/server_port_<*> = PROT=SMTP,PORT=<port>

is/SMTP/virt_host_0 = *:*;



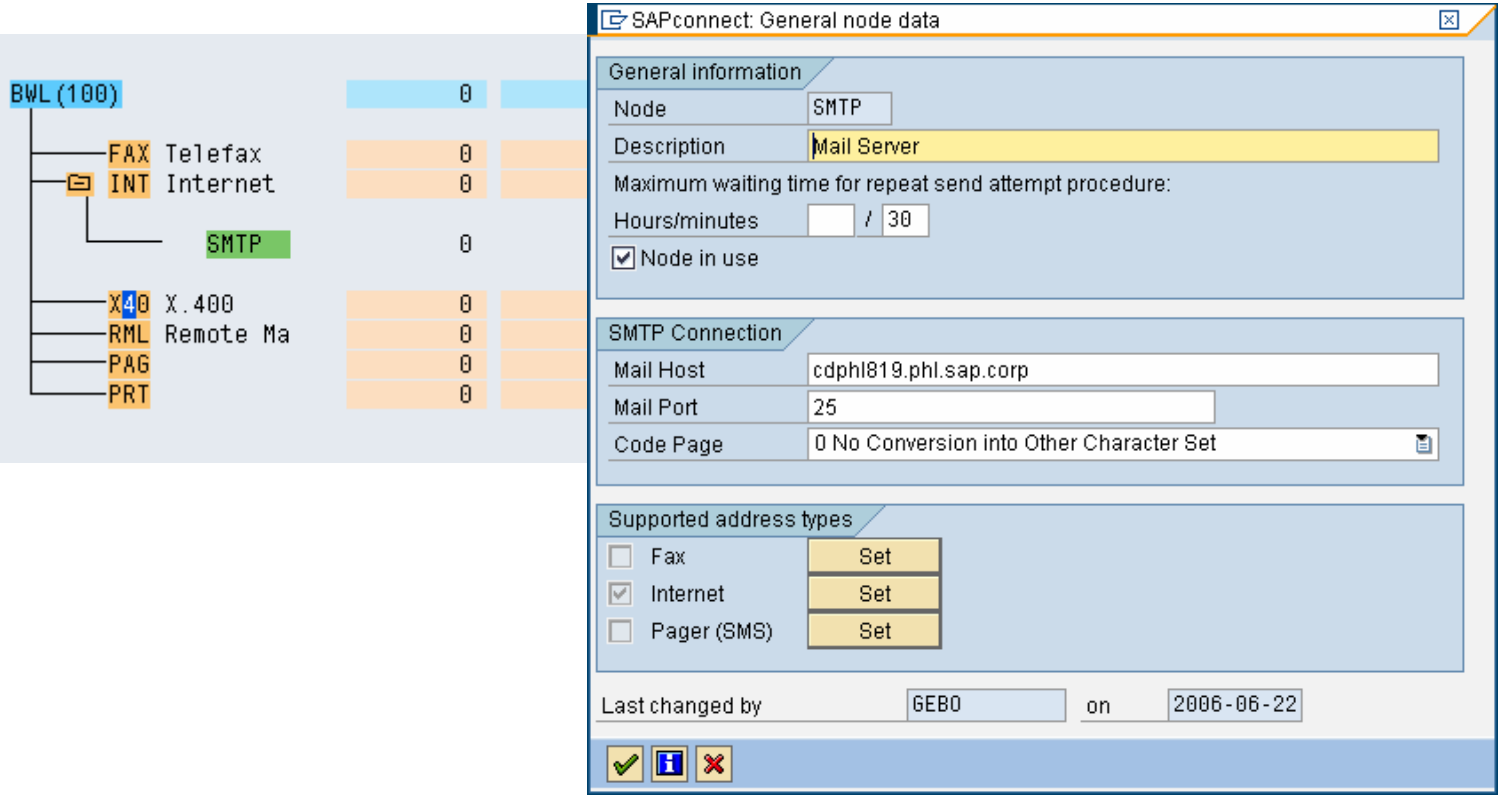
SAPconnect – Connecting SAP to the Outside World



SAPconnect – Configuring SMTP

To configure SAPconnect for receiving and sending emails, you must have a 3rd Party SMTP server already installed (example: Microsoft Exchange).

- **Transaction SCOT is where you can configure and monitor SAPconnect's SMTP connection to the SMTP server.**



Business Communication Services (BCS)

Business Communication Services are tools for interacting with email and workflow.

- **The Business Workplace can be accessed to view / send email and workflow items.**
- **More importantly for this class, there is a number of BCS classes that make it easy for use to Send email, and process received emails.**
  - **These classes include CL_BCS and CL_DOCUMENT_BCS which are heavily used to send and receive email.**

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Business Communication Services Classes

The Business Communication Services includes classes that can be used to create and send emails with attachments.

Classes that will be used:

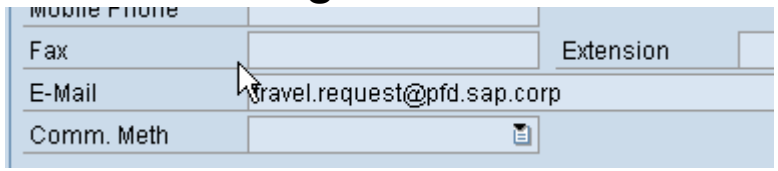
CL_BCS – class that actually send the email.

CL_DOCUMENT_BCS – class that represents the email document, and it's attachments.

CL_CAM_ADDRESS_BCS – used to create an email internet address of type IF_RECIPIENT_BCS.

CL_SAPUSER_BCS – class used to get the email address of an SAP user.

- **Note: All SAP users that will be used to send and receive email via SMTP must be given an email address.**



Programming Steps for Sending Email with Attachment

```
* (1) Create email subject and body
...
* (2) Create the email document object

...
* (3) Add Interactive Form/PDF attachment.

...
* (4) Create CL_BCS object used to send the email
...
* (5) Set the sender and recipient on the CL_BCS object
...
* (6) Send the email
...
```

* (1) Create email subject and body

```
* (1) Create email subject and body
  data: l_subject type so_obj_des,
        lt_bodytext type BCSY_TEXT,
        l_bodytext_row type soli.

  l_subject = 'Trip Creation Form'.

  concatenate 'Please fill out the form and return it to:'
              'travel.request@pdf.sap.corp'
              into l_bodytext_row.
  append l_bodytext_row to lt_bodytext.
```

* (2) Create the email document object

```
* (2) Create the email document object
  data: document TYPE REF TO cl_document_bcs,
        num_rows type i,
        textlength type so_obj_len.


  describe table l_mailtext lines num_rows.
  num_rows = num_rows * 255.
  move num_rows to textlength.

  document = cl_document_bcs=>create_document(
               i_type    = 'RAW'
               i_text    = l_mailtext
               i_length  = textlength
               i_subject = l_subject ).
```

* (3) Add Interactive Form/PDF attachment.

```
* Add attachment
data: attdoctype type soodk-objtp,
      atttitle   type SOOD-OBJDES,
      attsize    type SOOD-OBJLEN,
      pdftab     type SOLIX_TAB.

  attdoctype = 'pdf'.
  atttitle = 'CreateFlight'.
  attsize = xstrlen( pdf ).

  pdftab = cl_document_bcs=>xstring_to_solix(
             ip_xstring = pdf ).

  document->add_attachment( exporting I_ATTACHMENT_TYPE = attdoctype
                                      I_ATTACHMENT_SUBJECT = atttitle
                                      I_ATTACHMENT_SIZE = attsize
                                      I_ATTACHMENT_LANGUAGE = sy-langu
                                      I_ATT_CONTENT_HEX = pdftab ).
```

* (4) Create CL_BCS object used to send the email

```
* Create persistent send request
  data:send_request TYPE REF TO cl_bcs.
  send_request = cl_bcs=>create_persistent( ).

* Add document to send request
  send_request->set_document( document ).
```

* (5) Set the sender and recipient on the CL_BCS object

```
* Get sender object
  data: sender TYPE REF TO cl_sapuser_bcs.
  sender = cl_sapuser_bcs=>create( 'TRAVELREQ' ).

* Add sender
  CALL METHOD send_request->set_sender
    EXPORTING i_sender = sender.

* Create recipient.
  data: recipient TYPE REF TO if_recipient_bcs.
  recipient = cl_cam_address_bcs=>create_internet_address(
                                              p_addr ).

* Add recipient with its respective attributes to send request
  send_request->add_recipient( exporting i_recipient = recipient ).
```

* (6) Send the email

```
* Set send immediately
  send_request->set_send_immediately( 'X' ).

* Send document
  send_request->send( ).

  COMMIT WORK.
```

Catch the BCS Exceptions

The BCS classes throw exceptions, make sure place the BCS class interaction in a try/catch block:

```
* Create and send the email.

data: bcs_exception type ref to cx_bcs.

try.

* Enter your email code here!

catch cx_bcs into bcs_exception.

  data: ex_msg type string.

  ex_msg = bcs_exception->get_text( ).

  write: 'Caught exception.', ex_msg.

endtry.
```

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

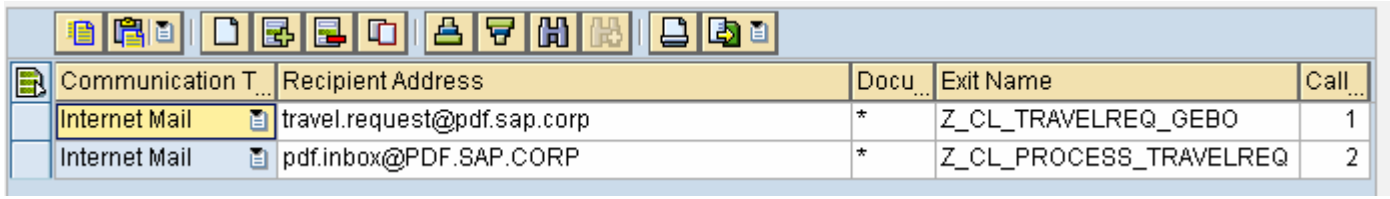
Processing Inbound Emails

Processing Interactive Forms

SAPconnect – Processing Inbound Email

SAPconnect allows the configuration of Inbound Processing rules.

- **If a rule is true for an incoming email, an Inbound Exit class can be executed to process the email and any attachments it may have.**
- **Rules are made up of communication type, recipient address, and document type.**
- **Multiple rules can be configured, each rule must be given a call sequence (a number). Inbound emails are checked against the lowest call sequence first (1, then 2, etc…).**
- **To create/modify/view rules, go to transaction SCOT, then menu path Settings > Inbound Processing.**



Communication T...	Recipient Address	Docu...	Exit Name	Call...
Internet Mail	travel.request@pdf.sap.corp	*	Z_CL_TRAVELREQ_GEBO	1
Internet Mail	pdf.inbox@PDF.SAP.CORP	*	Z_CL_PROCESS_TRAVELREQ	2

Inbound Process Classes

Inbound Exit classes are singleton classes that must implement the IF_INBOUND_EXIT_BCS interface.

- **Must implement the two methods defined on this interface.**
  - **CREATE_INSTANCE – contains code to create the singleton instance.**
    - **Must set the returning parameter RO_REF to the instance of the class.**
  - **PROCESS_INBOUND – contains code to process the inbound email.**
    - **Import parameter: IO_SREQ type ref to CL_SEND_REQUEST_BCS**
      - **Contains inbound email document class CL_DOCUMENT_BCS**
      - **Can be used to create reply emails.**
    - **Import parameter: IT_RECIPIENTS type BCSY_SMTPA – list of SMTP recipients.**
    - **Import parameter: IT_DOCTYPES type BCSY_SODOC – list of documents types make up the email.**
    - **Exporting parameter: E_RETCODE – used to control processing of other Inbound Exit processes in the call sequence.**
    - **Exporting parameter: ES_T100MSG – used to store T100 messages that can be written to a log.**

CREATE_INSTANCE Implementation

The CREATE_INSTANCE method is used to get a reference to the singleton instance of the class – if it does not exist it should create it:

```
* Check if the singleton instance has already
* been created.
  IF instance IS INITIAL.
    CREATE OBJECT instance.
  ENDIF.

* Return the instance.
  ro_ref = instance.
```

PROCESS_INBOUND Implementation

The PROCESS_INBOUND method is executed for every inbound email. You must place code here to do the email processing that is desired:

```
* Get the email document that was sent.

  data: document type ref to if_document_bcs.

  document = io_sreq->get_document( ).


* Get the interactive form attachment.

  data: pdf_table type BCSS_DBPC.

  pdf_table = document->get_body_part_content( 2 ).
```

Converting attachment type BCSS_DBPC to xstring

The attachment if returned from the CL_DOCUMENT_BCS object as a table of type BCSS_DBPC – to process an Interactive Form attachment it must be converted into an xstring type:

```
* Convert the pdf table into an xstring.

  data: pdf_xstring type xstring,

      pdf_line type solix.

  clear pdf_xstring.

  loop at pdf_table-cont_hex into pdf_line.

    concatenate pdf_xstring pdf_line-line into pdf_xstring
      in byte mode.

  endloop.
```

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Processing Interactive Forms

The processing of Interactive Forms is made up of two parts:

1. **Get the form data from the Interactive Form PDF document using the CL_FP_PDF_OBJECT class.**
   - **The form data is returned as an XML documnet.**
2. **Parsing the XML document using the iXML classes.**
   - **Once the form data is parsed from the XML document it can be used in your business processing (call a BAPI to create a flight booking for instance).**

Retrieving Form Data from Interactive Form (1)

```
* Get a reference to the form processing class.
  data: l_fp type ref to if_fp.
  l_fp = cl_fp=>get_reference( ).

* Get a reference to the PDF Object class.
  data: l_pdfobj type ref to if_fp_pdf_object.
  l_pdfobj = l_fp->create_pdf_object( ).

* Set the pdf in the PDF Object.
  l_pdfobj->set_document( pdfdata = pdf_xstring ).

* Set the PDF Object to extract data the Form data.
  l_pdfobj->set_extractdata( ).

* Execute call to ADS
  l_pdfobj->execute( ).
```

Retrieving Form Data from Interactive Form (2)

```
* Get the PDF Form data.

  data: pdf_form_data type xstring.

  l_pdfobj->get_data( importing formdata =
      pdf_form_data ).


* Convert the xstring form data to string so it can be
* processed using the iXML classes.

  data: converter type ref to cl_abap_conv_in_ce.


  converter = CL_ABAP_CONV_IN_CE=>CREATE( input =
      pdf_form_data ).

  converter->READ( importing data = formxml ).
```

Parsing the XML Document Using the iXML classes(1)

```
* Pull in the iXML type group.
  type-pools: ixml.

* Get a reference to iXML object.
  data: l_ixml type ref to IF_IXML.
  l_ixml = cl_ixml=>create( ).

* Get iStream object from StreamFactory
  data: streamFactory type ref to if_ixml_stream_factory.
  data: iStream type ref to if_ixml_istream.

  streamFactory = l_ixml->create_stream_factory( ).

  iStream =
      streamFactory->create_istream_string( formxml ).
```

Parsing the XML Document Using the iXML classes(2)

```
* Create an XML Document class that will be used to process the XML
  data: document type ref to if_ixml_document.
  document = l_ixml->create_document( ).

* Create the Parser class
  data: parser type ref to if_ixml_parser.
  parser = l_ixml->create_parser( stream_factory = streamFactory
                                  istream        = iStream
                                  document       = document ).
* Parse the XML
  parser->parse( ).
```

Parsing the XML Document Using the iXML classes(3)

```
* Define XML Node type object
  data: node type ref to if_ixml_node.

* Get the RESERVED Data Node and value.
  data: strChecked type string.
  node = document->find_from_name( name = 'RESERVED' ).
  strChecked = node->get_value( ).

* Look up the rest of the nodes and values.
  data: custbook type BAPISBONEW.
  node = document->find_from_name( name = 'AIRLINEID' ).
  custbook-airlineid = node->get_value( ).
  ...
```

Copyright 2005 SAP AG. All Rights Reserved

Copyright 2005 SAP AG. Alle Rechte vorbehalten

Colors

Remove page for slide show

R	0	239	77	191
G	51	159	77	191
B	102	0	77	191

R	124	87	190	80	33
G	30	129	102	34	87
B	31	174	0	71	71

Links to SAP PowerPoint Samples & Guidelines

Remove page for slide show

In order to access the links below, please view this slide in Slide Show mode

SAP_PPT Guidelines

Other than Title Slides and Divider Pages - The SAP PPT Guidelines include theory and examples of:

- Colors, Fonts & Bullets
- Image Usage & Quotes
- Graphics, Charts, Tables & Processes

SAP_PPT Samples

SAP PPT Sample pages include live ppt slides that were discussed in the SAP PPT Guidelines. Users can select from a library of slides including title slides, divider pages, bulleted slides, charts and graphs

For questions regarding the information and materials contained in these guidelines, please email branding@SAP.com