

DOE's Attachment Capability - with a Sample Application



Applies to:

SAP NetWeaver Mobile 7.1 SP03 and above, DOE (Data Orchestration Engine)

Summary

This document illustrates a sample application that uses the attachment (BLOB) handling capability of the DOE.

Author: Rohit Bhatnagar

Company: SAP labs India Pvt Ltd.

Created on: 26 October 2009

Author Bio



I work in the development team for SAP NetWeaver Mobile, and have been associated with this product for the last 2.2 years. My areas of expertise includes DOE Work-bench, Design time, and Backend Integration runtime.

Table of Contents

1. Introduction	3
2. Backend Related implementation	3
2.1 Backend schema & Backend Application	3
2.2. BAPI Wrappers	6
3. Data Object Modeling in DOE workbench	7
4. Backend Adapter:.....	8
5. Distribution Model:	10
6. Device Creation.....	11
7. Performance optimization (Compare memo configuration).....	12
Related Content	13
Copyright	14

1. Introduction

Application to display Employee details which includes BLOB data (Image) on a Laptop client using DOE middleware.

Definitions you need to know:

Binary Large Object: A database field that holds any digitized information, including text, images, audio or video. Also known simply as a "large object" or LOB, a BLOB may have a huge storage capacity.

Character Large Object: A database field that holds a large amount of text (character data). Also known as a "memo field" in some database programs. A DBCLOB is a "Double Byte" CLOB that supports two-byte Unicode characters

2. Backend Related implementation

2.1 Backend schema & Backend Application

To demonstrate an example lets consider only one table as Employee. For employee details we have employee name, company, address, age and image which we want to distribute to devices. Images will be distributed as attachment

First create one table as shown below:

Dictionary: Display Table

Field	Key	Initi...	Data element	Data Type	Length	Decim...	Sh
EMPLOYEE_NAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CHAR	32	0	
COMPANY	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	32	0	
EMPLOYEE_ADDRESS	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	32	0	
EMPLOYEE_AGE	<input type="checkbox"/>	<input type="checkbox"/>		INT4	10	0	
PICTURE	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	1	0	

Here Picture is declared as CHAR1 because this field will indicates weather there is an attachment or not for that employee record based on 'X' or " (Space) value.

Since there are two RFC function calls(GetDetail & GetAttachment) for every key, GetAttachment function call can be avoided for those keys by making the attachment fields(Char1 field) SPACE.

Now create one more table with only key of above table and BLOB (image) data.

Dictionary: Display Table

Field	Key	Initi	Data element	Data Type	Length	Decim	Sh
EMPLOYEE_NAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Expand include	CHAR	32	0	
PICTURE	<input type="checkbox"/>	<input type="checkbox"/>		RAWSTRING	0	0	

Picture is declared as RAWSTRING.

For loading data create a simple backend application,

SAP

Backend Application For BLOB Demo

Employee Name

Company

Address

Age

Picture

For persisting picture we need to convert Binary object to Xstring PAI Function module of application screen may look like this:

```
REPORT ZDEMO_EMPLOYEE_BLOB.
tables:zdemo_employee.

data: file_path_pic type string.
data: wa_employee type zdemo_employee.
data : wa_emp_attach type zdemo_emp_attach.
DATA file_table type filetable.
data wa_filetable like line of file_table.

MODULE USER_COMMAND_1000 INPUT.
  data:gfgcode like sy-UCOMM.
  gfgcode = sy-ucomm.
  CASE gfgcode.
    WHEN 'BACK'.
      set screen 0.
    WHEN 'SAVE'.
      "save employee profile data
      perform save_data.
    WHEN OTHERS.
```

```

ENDCASE.
ENDMODULE.                " USER_COMMAND_1000  INPUT
*&-----*
*&      Form  SAVE_DATA
*&-----*
*      text
*-----*
* -->  p1      text
* <--  p2      text
*-----*
FORM SAVE_DATA .

  select single * from ZDEMO_EMPLOYEE into wa_employee where employee_name = ZDEMO
_EMPLOYEE-EMPLOYEE_NAME.
  if sy-subrc <> 0.
    wa_employee-EMPLOYEE_NAME = ZDEMO_EMPLOYEE-EMPLOYEE_NAME.
    wa_employee-COMPANY = ZDEMO_EMPLOYEE-company.
    wa_employee-EMPLOYEE_ADDRESS = ZDEMO_EMPLOYEE-EMPLOYEE_address.
    WA_EMPLOYEE-EMPLOYEE_AGE = ZDEMO_EMPLOYEE-EMPLOYEE_age.
    if file_path_pic is not initial.
      wa_employee-PICTURE = 'X'.
    endif.
    insert zdemo_employee from wa_employee.
    " now to save attachments
    perform save_attachments.
    message 'Record saved!' type 'S'.
  else.
    message 'Username Profile Already exists' type 'S' display like 'E'.
  endif.
ENDFORM.                " SAVE_DATA
*&-----*
*&      Form  SAVE_ATTACHMENTS
*&-----*
*      text
*-----*
* -->  p1      text
* <--  p2      text
*-----*
FORM SAVE_ATTACHMENTS .
  types: begin of ty_itab,
         raw(255) type x,
         end of ty_itab.
  data itab type table of ty_itab.
  data length type i.
  if wa_employee-PICTURE = 'X'.

    CALL FUNCTION 'GUI_UPLOAD'
      EXPORTING
        FILENAME = file_path_pic
        FILETYPE = 'BIN'
      IMPORTING
        FILELENGTH = length
      TABLES
        DATA_TAB = itab.

    wa_emp_attach-employee_name = zdemo_employee-employee_name.

```

```

CALL FUNCTION 'SCMS_BINARY_TO_XSTRING'
  EXPORTING
    INPUT_LENGTH = length
  IMPORTING
    BUFFER       = wa_emp_attach-picture
  TABLES
    BINARY_TAB   = itab.

  insert Zdemo_emp_attach from wa_emp_attach.
endif.
ENDFORM.                    " SAVE_ATTACHMENTS

```

2.2. BAPI Wrappers

Now create GETLIST, GETDETAILS BAPIWRAPPERS and attachment Function module

For download attachment function module the importing parameter will be the key fields of the node

And exporting parameter will be the BLOB/CLOB field.

```

FUNCTION ZDEMO_EMPLOYEE_GETLIST.
  *"-----
  *"*"Local Interface:
  *"  TABLES
  *"    RETURN STRUCTURE BAPIRET2
  *"    IT_ZDEMO_EMPLOYEE STRUCTURE ZDEMO_EMPLOYEE
  *"-----
  select * from ZDEMO_EMPLOYEE into table IT_ZDEMO_EMPLOYEE.
ENDFUNCTION.
FUNCTION ZDEMO_EMPLOYEE_GETDETAILS.
  *"-----
  *"*"Local Interface:
  *"  IMPORTING
  *"    VALUE (EMPLOYEE_NAME) TYPE ZDEMO_EMPLOYEE-EMPLOYEE_NAME
  *"  EXPORTING
  *"    VALUE (WA_EMPLOYEE_NAME) TYPE ZDEMO_EMPLOYEE
  *"  TABLES
  *"    RETURN STRUCTURE BAPIRET2
  *"-----
  select single * from ZDEMO_EMPLOYEE into wa_employee_name where
  employee_name = employee_name.
ENDFUNCTION.
FUNCTION ZDEMO_ATTACHMENT_PIC.
  *"-----
  *"*"Local Interface:
  *"  IMPORTING
  *"    VALUE (EMPLOYEE_NAME) TYPE ZDEMO_EMPLOYEE-EMPLOYEE_NAME
  *"  EXPORTING
  *"    VALUE (EX_PICTURE_ATTCH) TYPE ZDEMO_EMP_ATTACH-PICTURE
  *"  TABLES
  *"    RETURN STRUCTURE BAPIRET2
  *"-----
  data wa_attach type ZDEMO_EMP_ATTACH.

  select single * from ZDEMO_EMP_ATTACH into wa_attach where EMPLOYEE_NAME = EMPLOYEE_NAME.

```

```
ex_picture_attach = wa_attach-PICTURE.  
ENDFUNCTION.
```

3. Data Object Modeling in DOE workbench

- i. Go to SDOE_WB Transaction
- ii. Create an SWCV and a data object having only one node (root node) to contain employee profile as well as image.
- iii. For the field corresponding to BLOB, we need to select the check box corresponding to Binary memo. Only if this field is marked as "binary memo" will DOE consider it for attachment relevance.

Employee_details node:

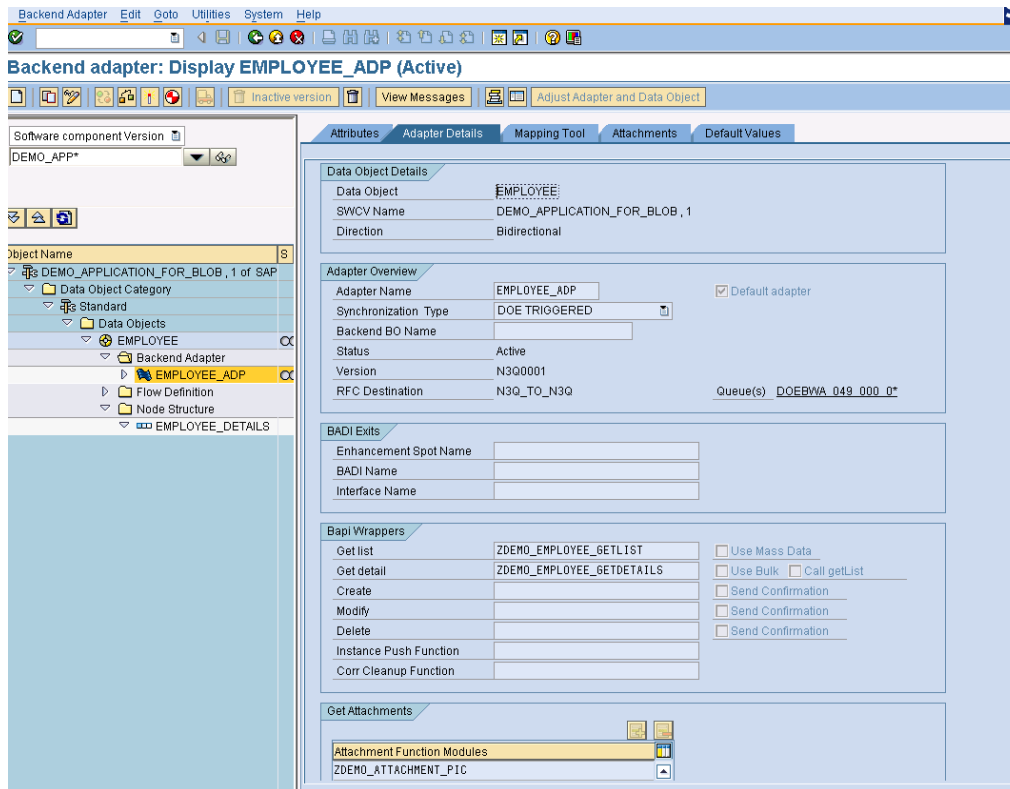
Data Object :Display EMPLOYEE (Active)

Position	Field Name	Data element	Data Type	Length	Decimal	Description
1	SYNCKEY_MW	CHAR32	CHAR	32	0	Character field, length 32
2	EMPLOYEE_NAME		CHAR	32	0	
3	EMPLOYEE_AGE		INT4	10	0	Natural number
4	PICTURE		RAWSTRING	0	0	
5	COMPANY		CHAR	32	0	
6	EMPLOYEE_ADDRESS		CHAR	32	0	

- iv. Now activate the data object and create Backend Adapter for it.
- v. Here the activation of data object will create a different table for BLOB similar to CDS table.
- vi. The name of the table will be of following pattern SDOE*_BM (and SDOE*_TM in case of text memo)

4. Backend Adapter:

- i. Right click on active Data Object and create Backend adapter.
- ii. Give the required details viz : name, synchronization type, bapiwrappers and Attachment function module.



- iii. Now click on mapping tool tab and do the mapping. Here the field "picture" will not be mapped with automatic mapping because field "picture" in getdetails bapiwrapper is of type CHAR1. Therefore we will have to map it explicitly using left arrow button. The below picture represent this:

5. Distribution Model:

For my simple application I have defined a simple DM on the data object with a bulk rule to distribute the content to all the subscribing devices.

- 1- Create a Distribution model by right clicking on data object. Create a bulk rule to distribute it to all the devices.
- 2- Activate the DM
- 3- The DM looks as following:

Distribution Model: EMPLOYEE_DM (Active)

The screenshot shows the SAP Distribution Model configuration for 'EMPLOYEE_DM'. The interface includes a toolbar with actions like 'Add Rule', 'Add Dependency', 'Add Bulk Rule', 'Add Dependency Bulk Rule', 'Inactive Version', and 'Print'. The left pane shows a tree view of the application structure, with 'EMPLOYEE_DM' selected under 'Distribution Model'. The main pane displays the following details:

Distribution Model Name:	EMPLOYEE_DM
Distribution Model Description:	Distribution model for employee
Distribution Model SWCV:	DEMO_APPLICATION_FOR_BLOB , 1 of SAP
Data Object Name:	EMPLOYEE
Data Object SWCV:	DEMO_APPLICATION_FOR_BLOB , 1 of SAP
Version:	N3Q0001
Status:	Active

Below the details, the 'Rules' tab is active, showing a table of rules:

Rule Name	Description	Type of Rule
EMPLOYEE_BULKRULE	bulk rule to distribute emp details	Bulk Rule

Activation of Rule:

1. Go to Admin portal
2. Click on Administration
3. Click on Distribution Rule Administration
4. Specify SWCV in Distribution Model SWCV
5. Click on filter
6. select the rule and activate it

The screenshot displays the SAP NetWeaver Data Orchestration Engine interface. The top navigation bar includes links for Mobile Overview, Administration, Configuration, Monitoring, and Statistics. The left sidebar contains a tree view with categories like Device Management, Hierarchy Groups, Software Package Management, Agents Administration, Distribution Rule Administration, and Role Management. The main content area shows the 'Device Management' configuration for a device named 'DEMO_BLOB'. Key fields include Device Id (0019BB3E4CD002DC83F846AF27E857EF), Status (Initial), and Queue Name (MMV_Q_000000000000004119). Below these fields is a table for 'Manage DMSWCVs' with columns for Operational status, DMSWCV name, and a checkbox. The table contains three rows: one for 'DEMO_APPLICATION_FOR_BLOB, 1 of SAP' (checked), one for 'SAP BASIS 7.10' (unchecked), and one for 'DEMO_APPLICATION_FOR_BLOB, 1 of SAP' (checked).

10. Go to device status and trigger pending extract

11. In the device out bound queue, check the content of corresponding field of the respective message. Being binary the content may not be readable.

7. Performance optimization (Compare memo configuration)

This configuration parameter is used to compare memo fields explicitly even if backend says there is an update on memo fields this will be useful because download of full blob will not be needed in case of false update.

1. Go to configuration
2. click on backend configuration
3. select COMPARE_MEMO field
4. Insert SWCV and Data object name.
5. Click on save.

Related Content

For more information, visit the [Mobile homepage](#).

Copyright

© 2009 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, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

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 Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, 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 in several other countries all over the world. 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.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.