Creating users in Active Directory from employee data stored in SAP HR

Andre Fischer, Project Manager CTSC - MS, SAP AG

Summary

SAP provides an interface that allows creating and modifying users in a LDAP directory server such as Microsoft Active Directory from employee data stored in SAP HR as part of the SAP standard. The interface called HR Data Retrieval in a LDAP Enabled Directory Service extracts data using a query or an ABAP report and performs the export into a LDAP directory server using the LDAP Connector. The usage of this interface allows for an optimized administration of users and thus can reduce the operational costs of an IT landscape. This collaboration brief describes how the LDAP interface has to be configured for Microsoft Active Directory and provides a sample report that can be used for testing purposes.

Applies to

- mySAP HR 4.6C together with a SAP Web Application Server 6.20 and higher
- SAP R/3 Enterprise, mySAP ERP
- Microsoft Active Directory 2000 and 2003

Keywords

Active Directory, HR, LDAP

Level of difficulty

Technical consultants, Developers

Contact

For feedback or questions you can contact the Collaboration Technology Support Center at ctsc@sap.com. Please check the .NET interoperability area in the SAP Developer Network http://www.sdn.sap.com/sdn/developerareas/dotnet.sdn for any updates or further information.
Copyright 2004 SAP AG. All rights reserved.

All other product and service names mentioned are the trademarks of their respective companies

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, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation 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.
# Contents

Summary .................................................................................................................. 1  
Applies to .................................................................................................................. 1  
Keywords .................................................................................................................... 1  
Level of difficulty ..................................................................................................... 1  
Contact ....................................................................................................................... 1  
Contents ..................................................................................................................... 3  
Introduction .............................................................................................................. 4  
Step 1: Data extraction in SAP HR ........................................................................ 5  
Step 2: Configuration of the LDAP interface .......................................................... 7  
  RFC destination and LDAP Connector ........................................................................ 7  
  Press the button Connector ...................................................................................... 9  
  LDAP Connector ..................................................................................................... 9  
  System logon ......................................................................................................... 10  
  LDAP server ......................................................................................................... 11  
  Attribute mapping ................................................................................................. 13  
Step 3: BADI Implementation .................................................................................. 14  
Test report Z_TEST_SPLDAP_RECEIVE ................................................................. 15  
Result ......................................................................................................................... 16  
User information in Active Directory ..................................................................... 18  
  Attributes that can be provided by mySAP HR ...................................................... 18  
  Attributes that are typically maintained in Microsoft Active Directory ............... 18  
Conclusion ............................................................................................................... 18  
Limitations ............................................................................................................... 18  
Troubleshooting ..................................................................................................... 19  
References .............................................................................................................. 19  
Appendix 1: Test report for SPLDAP_RECEIVE_ATTRIBUTES .............................. 20  
Appendix 2: Test report SAP HR data extraction Z_SAP_HR_LDAP ..................... 22
**Introduction**

Using the SAP HR LDAP interface companies can set up automated processes to create and update users in Active Directory from employee data stored in SAP HR.

The SAP HR system is the master for basic user data such as
- First Name
- Last Name
- Employee Number
- …

Since user data has not to be entered manually the correctness of user data is enhanced and the process of changing user data is speed up.

The SAP HR LDAP interface uses the LDAP Connector. The ABAP coding used by the SAP HR LDAP interface is shipped only with SAP basis release 6.x onwards. If employee data is to be extracted from a SAP HR 4.6 system the appropriate LDAP function modules have to be called remotely in a separate Web Application Server which then acts as a LDAP gateway.

---

**Data export from mySAP HR using LDAP interface**

If SAP HR is running on a SAP Enterprise System or higher the function module `SPLDAP_RECEIVE_ATTRIBUTES` can be called locally.

The data from SAP HR can be extracted using two different methods. The first method is using a query while the second method is based on an extraction report.
The extraction report is the method of choice if complicated data is to be extracted from SAP HR. Furthermore it is easy to format the data before it is exported to the Active Directory. A typical example is that the telephone number in SAP HR has a different format than the one used in Active Directory.

**Step 1: Data extraction in SAP HR**

The extraction report in the appendix is based on the SAP report RPLDAP_MANAGER. It uses the logical database PNP. The macro RP_PROVIDE_FROM_LAST is used to retrieve the last entry of the current period in the table header entry from an internal infotype table (here p0001 and p0002). The data is transmitted to the function module SPLDAP_RECEIVE_ATTRIBUTES.

```abap
CALL FUNCTION 'SPLDAP_RECEIVE_ATTRIBUTES' 
  DESTINATION LDAPDEST 
  EXPORTING 
    LOGSYS = LOGSYS 
    SERVERID = LDAPSRV 
    ATTRIBUTES_S = attributes[] 
    INITIAL_RUN = LDAPINITIALRUN 
  IMPORTING 
    RETURN = ERRORS[].
```

The function module SPLDAP_RECEIVE_ATTRIBUTES is part of the ABAP stack in a Web Application Server and is remote enabled. If the extraction report runs in a SAP HR system having the release 4.6 or lower it can to be called remotely in separate SAP Web Application Server that than acts as a LDAP gateway.

The function module SPLDAP_RECEIVE_ATTRIBUTES needs the following input parameters:

- **DESTINATION** RFC destination that is configured to access the SAP Web Application Server remotely where the LDAP Connector is configured. (Only needed for a SAP HR system having a release of 4.6 or lower)
- **LOGSYS** Logical system name of the client where the extraction report runs. This value is retrieved using the function module ‘OWN_LOGICAL_SYSTEM_GET’
- **SERVERID** Name of the LDAP server as it is configured in transaction LDAP in the SAP Web Application Server
- **ATTRIBUTES_S** Internal table that receives the name and values of the Logical SAP Data Fields that are mapped in transaction LDAP against the directory services attributes of the user object. The internal table has the following fields:
  - PERNR
  - ATTR_TAB
If this flag is set the function module first tries to create a user. It will try to update the user if the user already exists. If the flag is omitted the function module will first try to update the user and will then try to create the user if it does not exist.

The logical SAP data fields can be represented by freely-definable names whereas the name of the SAP data structure is fixed to EMPLOYEE. In our example we therefore choose meaningful names such as FIRSTNAME and LASTNAME for the SAP data fields. The SAP data fields are mapped to the directory services attributes using transaction **LDAPMAP** in the SAP Web Application Server as described later.

If the first name and the last name of the two employees Bill Smith and Bob Smith are extracted the table **ATTRIBUTES_S** will have the following content.

<table>
<thead>
<tr>
<th>PERNR</th>
<th>ATTR_TAB</th>
<th>ATTR_FIELD</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000001</td>
<td>EMPLOYEE</td>
<td>FIRSTNAME</td>
<td>Bill</td>
</tr>
<tr>
<td>0000001</td>
<td>EMPLOYEE</td>
<td>LASTNAME</td>
<td>Smith</td>
</tr>
<tr>
<td>0000001</td>
<td>EMPLOYEE</td>
<td>SAMACCOUNTNAME</td>
<td>E00000001</td>
</tr>
<tr>
<td>0000002</td>
<td>EMPLOYEE</td>
<td>FIRSTNAME</td>
<td>Bob</td>
</tr>
<tr>
<td>0000002</td>
<td>EMPLOYEE</td>
<td>LASTNAME</td>
<td>Smith</td>
</tr>
<tr>
<td>0000002</td>
<td>EMPLOYEE</td>
<td>SAMACCOUNTNAME</td>
<td>E00000002</td>
</tr>
</tbody>
</table>

The filling of the internal table is performed in the following section of the coding.

```plaintext
get pernr.
  rp-provide-from-last p0001 space keyda keyda.
  rp-provide-from-last p0002 space keyda keyda.
  ATTRIBUTES_WA-PERNR = p0001-pernr.

  * lastname
    attributes_wa-attr_tab = 'EMPLOYEE'.
    attributes_wa-attr_field = 'LASTNAME'.
    attributes_wa-value = p0002-nachn.
    append attributes_wa to attributes.

  * firstname
    attributes_wa-attr_tab = 'EMPLOYEE'.
    attributes_wa-attr_field = 'FIRSTNAME'.
    attributes_wa-value = p0002-vorna.
    append attributes_wa to attributes.

  * SAMAccountName
    attributes_wa-attr_tab = 'EMPLOYEE'.
    attributes_wa-attr_field = 'SAMACCOUNTNAME'.
```

* --------------------------------
* Using the employee number a unique name is created
* for the sAMAccountName
* --------------------------------
concatenate 'E' p0001-pernr into attributes_wa-value.
append attributes_wa to attributes.

* other attributes have to added here.
* attributes_wa-attr_tab = 'EMPLOYEE'.
* attributes_wa-attr_field = '<name of SAP data field>'.
* <coding to retrieve data from SAP HR if it cannot be retrieved>
* <from a infotype or subtype>
* attributes_wa-value = <SAP HR data>.
* append attributes_wa to attributes.
end-of-selection.

If additional data such as the cost center should be exported from SAP HR than one would just have to add the following coding.

* cost center
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'COSTCENTER'.
attributes_wa-value = p0001-kostl.
append attributes_wa to attributes.

---

**Step 2: Configuration of the LDAP interface**

In the following we will describe the configuration steps that are necessary for the configuration of the LDAP Connector. This configuration steps have to be performed on the SAP Web Application Server.

**RFC destination and LDAP Connector**

First you have to create a RFC destination in the Web Application Server that is used for the communication with the LDAP Connector using transaction SM59. The LDAP Connector has to be configured using connection type T with activation type “Registered Server Program”. As “ProgramID” speficy the same string as for the destination name, which in turn should follow the naming convention

LDAP_<ApplicationServerName>[_<n>]

(use the postfix number if you configure more than one LDAP Connector to run on the same application server).
As gateway options enter the gateway of the application server where the LDAP Connector will be started.
When the RFC connection has been created the LDAP Connector can be configured using transaction LDAP.

Figure 1: Configuration of the RFC destination of the LDAP Connector
Press the button Connector.

LDAP Connector

The next configuration step is the definition of the LDAP Connector. Start transaction LDAP and choose the button Connector.
The connector name is the RFC destination that has been created in the previous step.

![Figure 3: LDAP Connector configuration](image)

**System logon**

As a next step the credentials have to be specified that are used for the connection to the LDAP directory as the so called *System Logon* (here SAPHRLDAP). The user is specified using its *distinguishedName* while the password is stored in the secure storage.

Start transaction LDAP and choose the button *System Users*. 
The distinguished Name can be determined from Active Directory using tools like ADSIEdit.msc.

**LDAP server**

Now we can specify the configuration settings for the LDAP server including the System Logon that should be used for the LDAP connection. Start transaction LDAP and choose the button *Server Names.*
Figure 5: LDAP server configuration

To check the configuration start transaction LDAP -> Log on. Check the check box *Use System User.*

Figure 6: Log on to Directory Service using the System User
Attribute mapping

The employee data that is retrieved from the SAP HR system is sent to the SAP Web Application Server using the data structure EMPLOYEE and self defined SAP data fields.

The attribute mapping allows that the values of the SAP data fields are mapped to the corresponding attribute names used in Active Directory. The attribute mapping can also be called directly using transaction LDAPMAP. The mapping flags have the following meaning:
<table>
<thead>
<tr>
<th>Filter</th>
<th>Determines how corresponding entries for SAP Objects can be found in the directory (Only one line can be checked)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import Mapping</td>
<td>Determines which mappings are used to read directory entries (not relevant for this scenario. Only for the “Filter” line this indicator must be set)</td>
</tr>
<tr>
<td>Export Mapping</td>
<td>Determines which mappings are used to write directory entries</td>
</tr>
<tr>
<td>Required</td>
<td>Determines which attributes are essential (mandatory) for new directory entries</td>
</tr>
<tr>
<td>RDN mapping</td>
<td>Marks the mapping which is used to form the RDN of new directory entries (Only one line can be checked)</td>
</tr>
<tr>
<td>Import</td>
<td>Not relevant for this scenario</td>
</tr>
<tr>
<td>Export</td>
<td>Not relevant for this scenario</td>
</tr>
</tbody>
</table>

Remarks:

The object class that we use in the Active Directory (user) has two mandatory attributes which are `cn` (common name) and `sAMAccountName`. If these attributes are not used, the creation of an object using this object class will fail. Therefore we marked them as required.

The attribute `sAMAccountName` has to be unique in each domain and is indexed too. Therefore the flag `filter` is checked for this mapping.

The function module SPLDAP_RECEIVE_ATTRIBUTES internally fills the SAP data field `KEY` in the structure EMPLOYEE. Thus a mapping has to be maintained for the SAP data field `KEY`. Since this field is used to build the relative distinguished Name for the user object the flag `RDN Mapping` has to be checked for this mapping.

However no data has to be filled into the internal table ATTRIBUTES_S for the SAP data field `KEY` that is send to the function module.

As a default the function module fills the data field `KEY` with a default value that consists out of the name of the logical system of the SAP HR system and the personnel number) being used as the naming attribute `cn` in the Active Directory.

If another value than the default should be used for `cn` one has to implement the BADI `HRLDAP_ATTRIBUTES` in the SAP Web Application Server.

**Step 3: BADI Implementation**

The method `GET_RDN` of the BADI `HRLDAP_ATTRIBUTES` has to be implemented in the SAP Web Application Server. To do this start transaction SE19 enter a new name for
the implementation like \texttt{Z\_HRLDAP\_ATTRIBUTES} and follow the steps described in the documentation http://help.sap.com/saphelp_erp2004/helpdata/en/eb/3e7cf4940e11d295df0f000e82de14a/content.htm

In our example we fill the value of \texttt{cn} with the first name and the last name of an employee using a space as a separator. This is the default value that is used if a user object is created using the MMC SnapIn "Users and Computers".

```
method IF\_EX\_HRLDAP\_ATTRIBUTES\_GET\_RDN .

FIELD\_SYMBOLS: <fs\_data> LIKE LINE OF data.

DATA vals_wa TYPE valstructc.

CLEAR rdn.
CLEAR xrdn.

* create cn that has the following structure
* firstname space lastname
READ TABLE data
  WITH KEY var = 'EMPLOYEE'
  fld = 'FIRSTNAME'
  ASSIGNING <fs\_data>.

IF sy\_subrc EQ 0.
  READ TABLE <fs\_data>-vals INDEX 1
      INTO vals_wa.
  IF sy\_subrc EQ 0.
    rdn = vals_wa-val.
  ENDIF.
ENDIF.

READ TABLE data
  WITH KEY var = 'EMPLOYEE'
  fld = 'LASTNAME'
  ASSIGNING <fs\_data>.

IF sy\_subrc EQ 0.
  READ TABLE <fs\_data>-vals INDEX 1
      INTO vals_wa.
  IF sy\_subrc EQ 0.
    concatenate rdn vals_wa-val into rdn separated by space.
  ENDIF.
ENDIF.

* CONCATENATE logsys pernr INTO rdn SEPARATED BY space.
endmethod.
```

**Test report Z\_TEST\_SPLDAP\_RECEIVE**

The settings in the LDAP configuration and the BADI implementation mentioned above can be tested using the report \texttt{Z\_TEST\_SPLDAP\_RECEIVE}. This way it is possible to test the technical side of the SAP HR LDAP interface without the need to have development authorizations in the SAP HR system.
Result

While new users are created as deactivated accounts without a password in Active Directory existing user accounts will be updated when user information changes in SAP HR.
Figure 10: Telephone number user Bill Smith

Figure 11: sAMAccountName user Bill Smith
User information in Active Directory

User information in Active Directory is stored in the attributes of the user object. While for basic user information such as the first name or the last name the SAP HR system will be the leading system other attributes such as the email address might be maintained by the Exchange administration.

Attributes that can be provided by mySAP HR

distinguishedName: CN=Bill Smith,OU=SAP_HR,DC=MSCTSC,DC=SAP,DC=CORP
sn: Smith
givenName: Bill
employeeNumber: 00000001
sAMAccountName: E00000001
userPrincipalName: Bill.Smith@SAP.COM

Attributes that are typically maintained in Microsoft Active Directory

mail: Bill.Smith@sap.com
memberOf: CN=Users,DC=MSCTSC,DC=SAP,DC=CORP;
CN=Domain Admins,CN=Users,DC=MSCTSC,DC=SAP,DC=CORP;
CN=SAP Users,CN=Users,DC=MSCTSC,DC=SAP,DC=CORP;

Conclusion

SAP HR data can be used to control it-user life cycles. Using the SAP HR LDAP interface processes can be made more secure and faster. Furthermore the quality of data in the Active Directory can be improved.

Limitations

This concept only works for data extraction from HR and can not be used to write back any information from the LDAP server to the HR system. However it is possible to write such functionality yourself using the LDAP function modules provided by the SAP standard.

The interface does not provide an option out of the box only to export the data of those employees that have changed. However this can be done programmatically if one
creates a table that has the same structure as the internal table `attributes` to store the data that is exported so that the report can check whether the data that is to be exported has changed.

**Troubleshooting**

If the LDAP trace is activated for the LDAP Server in transaction LDAP a log file is generated that can be accessed using transaction ST11:

```
<table>
<thead>
<tr>
<th>Viewed</th>
<th>Name</th>
<th>Length</th>
<th>Creator</th>
<th>Last change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dev_w0</td>
<td>38762</td>
<td>Administ</td>
<td>10.88.2885 14:02:29</td>
</tr>
<tr>
<td></td>
<td>dev_w1</td>
<td>212814</td>
<td>Administ</td>
<td>10.88.2885 14:02:37</td>
</tr>
<tr>
<td></td>
<td>dev_LDAP_HG0T58KX138_0.trc</td>
<td>4896</td>
<td>Administ</td>
<td>10.88.2885 14:02:32</td>
</tr>
<tr>
<td></td>
<td>dev_w12</td>
<td>422686</td>
<td>Administ</td>
<td>10.88.2885 14:02:22</td>
</tr>
</tbody>
</table>
```

Figure 12: LDAP Developer Trace

**References**

- SAP documentation: “HR Data Retrieval in a LDAP Enabled Directory Service”
  [http://service.sap.com/security](http://service.sap.com/security) -> Media Library -> Literature & Brochures or directly via the following link
- SAP Online Help: “Provide the Last Entry in the Period”
- SAP Online Help: “Configuring the LDAP Connector”
- Steps to Activate BAdI Implementations
Appendix 1: Test report for SPLDAP_RECEIVE_ATTRIBUTES

The following coding can be used to test the function module S

```plaintext
REPORT Z_TEST_SPLDAP_RECEIVE.
DATA: mid  TYPE sy-msgid VALUE 'LDAPSYNC',
     mtype TYPE sy-msgty VALUE 'I',
     num  TYPE sy-msgno.

tables: LDAPSERVER.

data:   ldapinitialrun(1),
     LOGSYS LIKE TBDLS-LOGSYS,
     ERRORS LIKE BAPIRET2 OCCURS 0,
     ERRORS_WA LIKE BAPIRET2.

* structure for short ldap-attributes
* fieldnames must be equal to basis structure LDA_ATTR_L (for RFC-call)

TYPES: BEGIN OF TS_LDAP_ATTR_L,
     PERNR LIKE LDA_ATTR_L-PERNR,
     ATTR_TAB LIKE LDA_ATTR_L-ATTR_TAB,
     ATTR_FIELD LIKE LDA_ATTR_L-ATTR_FIELD,
     VALUE LIKE LDA_ATTR_L-VALUE,
END OF TS_LDAP_ATTR_L.

data: attributes type ts_ldap_attr_l occurs 0,
     attributes_wa type ts_ldap_attr_l.

SELECTION-SCREEN BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.
PARAMETERS: P_TEST default 'X' AS CHECKBOX.
SELECTION-SCREEN END OF BLOCK B1.

Parameters: LDAPSRV Default 'SAPHRLDAP' LIKE LDAPSERVER-SERVERID,
            LDAPDEST Default 'NONE' LIKE rfcdes-rfcdest.

* LIKE LDAPSERVER-BASE allows case sensitive input

parameters:
sn default 'Smith' LIKE LDAPSERVER-BASE,
 givenNam default 'Bill' LIKE LDAPSERVER-BASE,
 samacc default 'E00000001' LIKE LDAPSERVER-BASE,
 tel default '+49 6227 747474' LIKE LDAPSERVER-BASE.

* get own logical system
CALL FUNCTION 'OWN_LOGICAL_SYSTEM_GET' IMPORTING
```
OWN_LOGICAL_SYSTEM             = LOGSYS
EXCEPTIONS
OWN_LOGICAL_SYSTEM_NOT_DEFINED = 1
OTHERS                        = 2.

IF SY-SUBRC NE 0.
* TODO: Komprimierung sy-mandt: 3 -> 2 Stellen !!!
  CONCATENATE SY-SYSID SY-MANDT INTO LOGSYS.
ENDIF.

ATTRIBUTES_WA-PERNR = '01234567'.
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'LASTNAME'.
attributes_wa-value = sn.
append attributes_wa to attributes.

ATTRIBUTES_WA-PERNR = '01234567'.
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'FIRSTNAME'.
attributes_wa-value = givenNam.
append attributes_wa to attributes.

ATTRIBUTES_WA-PERNR = '01234567'.
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'SAMACCOUNTNAME'.
attributes_wa-value = samacc.
append attributes_wa to attributes.

ATTRIBUTES_WA-PERNR = '01234567'.
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'TELEPHONENUMBER'.
attributes_wa-value = tel.
append attributes_wa to attributes.

IF P_TEST = 'X'.
  EXIT.
ENDIF.

* ---------------------------------------------------------------------
* send attributes to ldap client
CALL FUNCTION 'SPLDAP_RECEIVE_ATTRIBUTES'  
  DESTINATION LDAPDEST  
  EXPORTING  
    LOGSYS         = LOGSYS  
    SERVERID       = LDAPSRV  
  *     ATTRIBUTES_S   = attributes  
  INITIAL_RUN    = LDAPINITIALRUN  
  ATTRIBUTES_L   = attributes  
  *     ATTRIBUTES_X   = TOTAL_ATTRS_X  
  IMPORTING  
    RETURN         = ERRORS[ ]  
.
IF NOT ERRORS[ ] IS INITIAL.
  READ TABLE ERRORS INDEX 1 INTO ERRORS_WA.
  MESSAGE ID mid TYPE mtype
    NUMBER ERRORS_WA-NUMBER
    WITH ERRORS_WA-MESSAGE_V1 ERRORS_WA-MESSAGE_V2  
    ERRORS_WA-MESSAGE_V3 ERRORS_WA-MESSAGE_V4.
ENDIF.
Appendix 2: Test report SAP HR data extraction

**Z_SAP_HR_LDAP**

The following coding can be used to extract employee data from an SAP HR system and export the data using a LDAP Connector that is running on a SAP Web Application Server.

If the SAP HR system is running on SAP Basis Release 6.20 or higher no separate Web Application Server is necessary. In this case the LDAP Connector can be configured in the system itself and the LDAP destination can be left blank or omitted.

When creating the report in the SAP HR system you have to maintain the logical database name in the program attributes. If you forget to maintain the name of the logical database you will get an error like “PERNR” is not defined for the current logical database at line ...

To maintain the logical database name in the menu of the ABAP editor choose:

Goto --> Attributes.

```abap
REPORT Z_SAP_HR_LDAP USING DATABASE PNP.

DATA:   PLVAR     LIKE OBJEC-PLVAR,
        OBJID     LIKE HROBJECT-OBJID,
        KEYDA     LIKE PLOG-BEGDA,
        P_OBJECTS LIKE HROBJECT OCCURS 0,
        P_OBJECTS_WA LIKE HROBJECT,
        S_OBJECTS LIKE HROBJECT OCCURS 0,
        S_OBJECTS_WA LIKE HROBJECT,
        I1001_ITAB  LIKE P1001 OCCURS 0 WITH HEADER LINE,
        I1001_ITAB2  LIKE P1001 OCCURS 0 WITH HEADER LINE,
        LDAPSERVER LIKE LDA_TYPES-LDAPSERVER,
        ldapinitialrun like lda_types-flag,
        LOGSYS LIKE TBDLS-LOGSYS,
        ERRORS LIKE BAPIRET2 OCCURS 0,
        ERRORS_WA LIKE BAPIRET2.

DATA:   mid  TYPE sy-msgid VALUE 'LDAPSYNC',
        mtype TYPE sy-msgty VALUE 'I',
        num  TYPE sy-msgno.
```

* &--------------------------------------------------------------------*
* & Z_SAP_HR_LDAP                                                      *
* &                                                                    *
* &--------------------------------------------------------------------*
* & written by Andre Fischer, CTSC-MS, SAP AG                         *
* &                                                                    *
* &--------------------------------------------------------------------*

```abap
YPES: BEGIN OF TS_LDAP_ATTR_L,
      PERNR LIKE LDAATTR_L-PERNR,
      ATTR_TAB LIKE LDAATTR_L-ATTR_TAB,
```
ATTR_FIELD LIKE LDA_ATTR_L-ATTR_FIELD,
    VALUE LIKE LDA_ATTR_L-VALUE,
END OF TS_LDAP_ATTR_L.

data: attributes type ts ldap attr l occurs 0,
    attributes_wa type ts ldap attr l.

infotypes: 0001, 0002.
tables: pernr, rfcdes.

SELECTION-SCREEN BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.
PARAMETERS: P_TEST default 'X' AS CHECKBOX.
SELECTION-SCREEN END OF BLOCK B1.

Parameters: LDAPSRV Default 'SAPHRLDAP' LIKE LDA_TYPES-LDAPSERVER,
    LDAPDEST Default 'LDAPDEST' LIKE rfcdes-rfcdest.

at selection-screen.

clear: p_objects[], s_objects[], attributes[].

CALL FUNCTION 'RH_GET_PLVAR'
    IMPORTING
        PLVAR    = PLVAR
    EXCEPTIONS
        no_plvar = 1
        OTHERS   = 2.
if sy-subrc <> 0.
    MESSAGE E015(HRLDAP).
endif.

KEYDA = sy-datum.

get pernr.

rp-provide-from-last p0001 space keyda keyda.
rp-provide-from-last p0002 space keyda keyda.

ATTRIBUTES_WA-PERNR = p0001-pernr.

* lastname
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'LASTNAME'.
attributes_wa-value = p0002-nachn.
append attributes_wa to attributes.

* firstname
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'FIRSTNAME'.
attributes_wa-value = p0002-vorna.
append attributes_wa to attributes.

* sAMAccountName
attributes_wa-attr_tab = 'EMPLOYEE'.
attributes_wa-attr_field = 'SAMACCOUNTNAME'.

* Using the employee number a unique name is created
* for the sAMAccountName
* --------------------------------
concatenate 'E' p0001-pernr into attributes_wa-value.
append attributes_wa to attributes.
* other attributes have to added here.

* for each additional attribute an appropriate field
* has to defined in the structure EMPLOYEE
* for example you can choose the following:
* attributes_wa-attr_tab = 'EMPLOYEE'.
* attributes_wa-attr_field = 'TELEPHONE'.

* if you want to transfer the telephone number
* of an employee from SAP HR to Active Directory
* in the web Application Server an appropriate mapping
* has to be defined using transaction LDAP for each new
* attribute (here called TELEPHONE).

end-of-selection.

* get own logical system
  CALL FUNCTION 'OWN_LOGICAL_SYSTEM_GET'
  IMPORTING
    OWN_LOGICAL_SYSTEM             = LOGSYS
  EXCEPTIONS
    OWN_LOGICAL_SYSTEM_NOT_DEFINED = 1
    OTHERS                         = 2.

  IF SY-SUBRC NE 0.
    * TODO: Komprimierung sy-mandt: 3 -> 2 Stellen !!!
      CONCATENATE SY-SYSID SY-MANDT INTO LOGSYS.
    ENDIF.

  loop at attributes into attributes_wa.
    write: / ATTRIBUTES_WA-PERNR , attributes_wa-attr_tab.
    write:  attributes_wa-attr_field , attributes_wa-value.
  endloop.

  IF P_TEST = 'X'.
    EXIT.
  ENDIF.

end.

* send attributes to ldap client
  CALL FUNCTION 'SPLDAP_RECEIVE_ATTRIBUTES'
  DESTINATION LDAPDEST
  EXPORTING
    LOGSYS         = LOGSYS
    SERVERID       = LDAPSRV
    ATTRIBUTES_S   = attributes[]
    INITIAL_RUN    = LDAPINITIALRUN
    ATTRIBUTES_L   = attributes[]
    ATTRIBUTES_X   = TOTAL_ATTRS_X[].
  IMPORTING
    RETURN         = ERRORS[].

  IF NOT ERRORS[] IS INITIAL.
    READ TABLE ERRORS WA INDEX 1 INTO ERRORS_WA.
    MESSAGE ID mid TYPE mtype
      NUMBER ERRORS_WA-NUMBER
      WITH ERRORS_WA-MESSAGE_V1 ERRORS_WA-MESSAGE_V2
      ERRORS_WA-MESSAGE_V3 ERRORS_WA-MESSAGE_V4.
  ENDIF.