Paper 377-2012

Automagically Herding 101 SAS® Users from Microsoft Active Directory to SAS Metadata

Stephen Overton, Zencos Consulting, Cary, NC, USA

ABSTRACT

Creating users within SAS metadata can become a tedious task for the system administrator, especially in large environments with strict compliance policies around user access to data through SAS. This paper will discuss a process and provide code to import users from Microsoft Active Directory into SAS metadata. A stored process will be the graphical interface for the security administrator to pick-and-choose what users will be imported. This process can be extended further to import group memberships. This paper will only focus on importing users. Using a stored process to selectively import users from Active Directory streamlines user creation in SAS metadata and provides a point and click method to minimize human error.

INTRODUCTION

Microsoft Active Directory is the most popular tool for managing user authentication in most corporate IT environments today and is included in most Windows Server operating systems. SAS ships with macros, which can extract user accounts from Active Directory and create user accounts in SAS metadata. Rather than maintain user accounts separately in SAS metadata, these macros simplify user management because user accounts and groups are centrally managed in Active Directory and imported into SAS. By default, these macros import and synchronize all accounts from Active Directory. In order to take advantage of these macros the SAS metadata server must use host authentication against the same Active Directory provider users are extracted from. Pluggable Authentication Modules (PAM) provide similar host authentication in UNIX environments. More information on host authentication is provided at the URL listed in the References section below.

For organizations with strict change control processes a more controlled method may be required that imports user accounts selectively. This paper presents a method that graphically selects specific users to import from Active Directory into SAS metadata. If user access is managed by a dedicated security team at an organization, user creation in SAS metadata could be managed by this easy-to-use process. The alternative is to create users manually in SAS Management Console® but this can become tedious and cumbersome, and requires special training. This paper assumes the reader understands the technologies in this section and has an advanced understanding of SAS programming.

USER IMPORT MACROS FROM SAS FOUNDATION®

The following macros are the core components used to import and synchronize user accounts from Active Directory to SAS metadata: %MDUIMPC , %MDUIMPLB , %MDUEXTR , %MDUCMP , %MDUCHGV , %MDUCHGLB. They are located in the following directory: [SAS Home]\SASFoundation\9.3\core\sasmacro.

SAS Foundation also ships with a sample program called IMPORTAD.SAS, which imports all users using these macros. It is located in the following directory: [SAS Home]\SASFoundation\9.3\core\sample. This program and macros are documented online through SAS Technical Support. The URL for this can be found in the References Section.

The IMPORTAD.SAS program is used as a model for the process described throughout the remainder of this paper. It is split into two pieces and customized in order to selectively import Active Directory users. Due to the length of the code, only specific changes made to this code will be described in following sections. Full sample code that is used in this paper can be found at the URL listed in the References section.

BUILDING THE IMPORT PROCESS

SAS stored processes are very powerful tools to perform any SAS task. With the Enterprise Business Intelligence platform, they can be executed from a Web browser, which provides a robust way to run a process. The import process is broken into two chained stored processes and described in the following steps.

USER SELECTION STORED PROCESS

1. Create a session to share data between stored processes
2. Define parameters to retrieve and store data

3. Extract all users from Active Directory and store in shared data library
4. Define macro and macro variables for writing custom HTML form
5. Write HTML form with submit button to execute next stored process
6. User selects one or multiple users which are passed to next stored process

USER IMPORT STORED PROCESS

1. Define options for writing users back to metadata
2. Remove users not selected from shared datasets
3. Load user data from shared datasets into SAS metadata
4. Output table of users imported and/or users not imported
5. Close shared session

To display a prompt with a list of users to import, SAS code is executed before the prompt is generated to retrieve all users from Active Directory. The current prompt framework in SAS does not allow SAS code to run prior to prompting for user selection. Therefore, two chained stored processes must be used in order to prompt for users to import.

WRITING THE USER SELECTION STORED PROCESS

Use this stored process to establish a stored process session and get the user names to import.

USING STORED PROCESS SESSIONS

Stored processes can be chained together to perform multiple tasks and display multiple web pages. Passing data between sessions can be accomplished through URL parameters, client cookies, or on the server using sessions. The stored processes demonstrated in this paper use a shared session to pass user names selected for import. Additional documentation on sessions can be found in the Integration Technologies Developer's Guide that is listed in the References section of this paper.

The following code must be submitted first to create the session. A library with the LIBREF of "SAVE" is defined automatically when the macro is submitted. Data stored in the SAVE library can be accessed by any stored process that uses the shared session ID.

```
%global _SESSIONID _THISSESSION SAVE_importlibref SAVE_ADExtIDTag;
%let rc=%sysfunc(stpsrv_session(create,300));
```

- _SESSIONID: macro variable represents the unique identifier for the stored process session.
- _THISSESSION: macro variable used in URL strings to execute the next stored process within the same session.
- SAVE_importlibref: macro variable that stores the library where Active Directory data will be stored.
- SAVE_ADExtIDTag: Descriptive tag for the data extracted from Active Directory. This is stored in SAS metadata.

Note: To pass macro variables between stored processes, "SAVE_" must be prepended to the macro variable name.

PARAMETERS TO EXTRACT AND STORE ACTIVE DIRECTORY DATA

The following macro variables are defined in the User Selection stored process to connect to the Active Directory server. The user account that is used to extract data from Active Directory must be a Domain Admin.

```
/* network name/address of the AD server */
%let ADServer = "192.168.56.250";
/* Port on which LDAP interface is listening. 389 is a standard port. */
%let ADPort   = 389;
/* Specify the Distinguished Name in the LDAP hierarchy where People searches begin.*/
%let ADPerBaseDN ="ou=SGF12,dc=SGF12,dc=LOCAL";
/* Specify the Distinguished Name in the LDAP hierarchy where Group searches begin. */
%let ADGrpBaseDN ="ou=SGF12,dc=SGF12,dc=LOCAL";
/* Userid which will connect to the AD server and extract the information.    */
%let ADBindUser = "sasextract";
```

```
/* Password for Userid above.  */
%let ADBindPW = "XXXXXXXXXX";
```

Note: Contact the system administrator to obtain the parameters above.

PARAMETERS TO STORE AND INTERPRET DATA CORRECTLY

The following library parameters are set to the SAVE library, which is shared between both stored processes. Defining the session above creates this library.

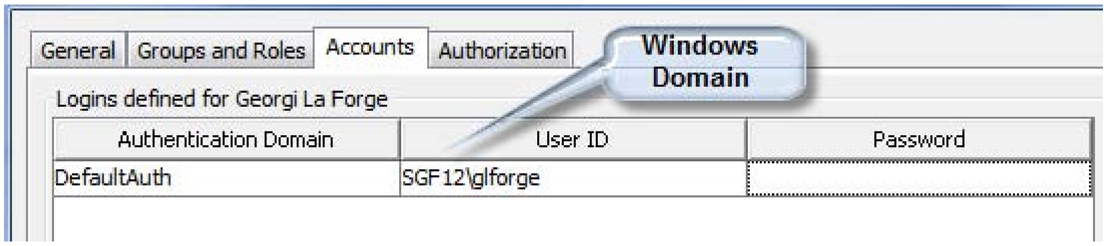
```
%let extractlibref=SAVE;

%let importlibref=SAVE;
```

For the Active Directory environment simulated for the stored process in this paper, the “distinguishedName” macro variable is used as the unique identifier for each user as shown below.

```
%let keyidvar=distinguishedName;
```

The Windows domain can be automatically set for each user imported into SAS metadata. This is useful for environments that require this to log in.

```
%let WindowsDomain=SGF12;
```



Authentication Domain	User ID	Password
DefaultAuth	SGF12\glforge	

Figure 1. Windows Domain automatically set in metadata user account

EXTRACTING ACTIVE DIRECTORY DATA THROUGH LDAP QUERIES

Retrieving data from Active Directory is performed using multiple LDAP queries. The LDAP queries originally written in the IMPORTAD.SAS code will need to be modified or confirmed to properly query the Active Directory structure of the environment it is run within. Incorrect queries will result in no data or errors. An LDAP query tool is useful in determining the structure of the Active Directory environment.

In the data step that creates the “ldapusers” table, the “filter” variables must be customized to the Active Directory structure. The following represents the changes made for the User Selection stored process in this paper.

```
filter="(&(&(displayName>=U)(displayName<=W))  (distinguishedName=*)  )";
%ldapextrpersons
```

The majority of the following code in the IMPORTAD.SAS program is copied into the stored process and is used to extract additional user and group data.

BUILDING A CUSTOM HTML USER SELECTION FORM

In order to prompt for a user list to import into SAS metadata, a custom HTML form has to be written. A multiple selection list is used to allow the user to import more than one user at a time. This form is very simple for demonstration purposes but could be extended to a higher functioning input form using jQuery or any other JavaScript plugin. A screenshot of the HTML input form is shown in Figure 2.

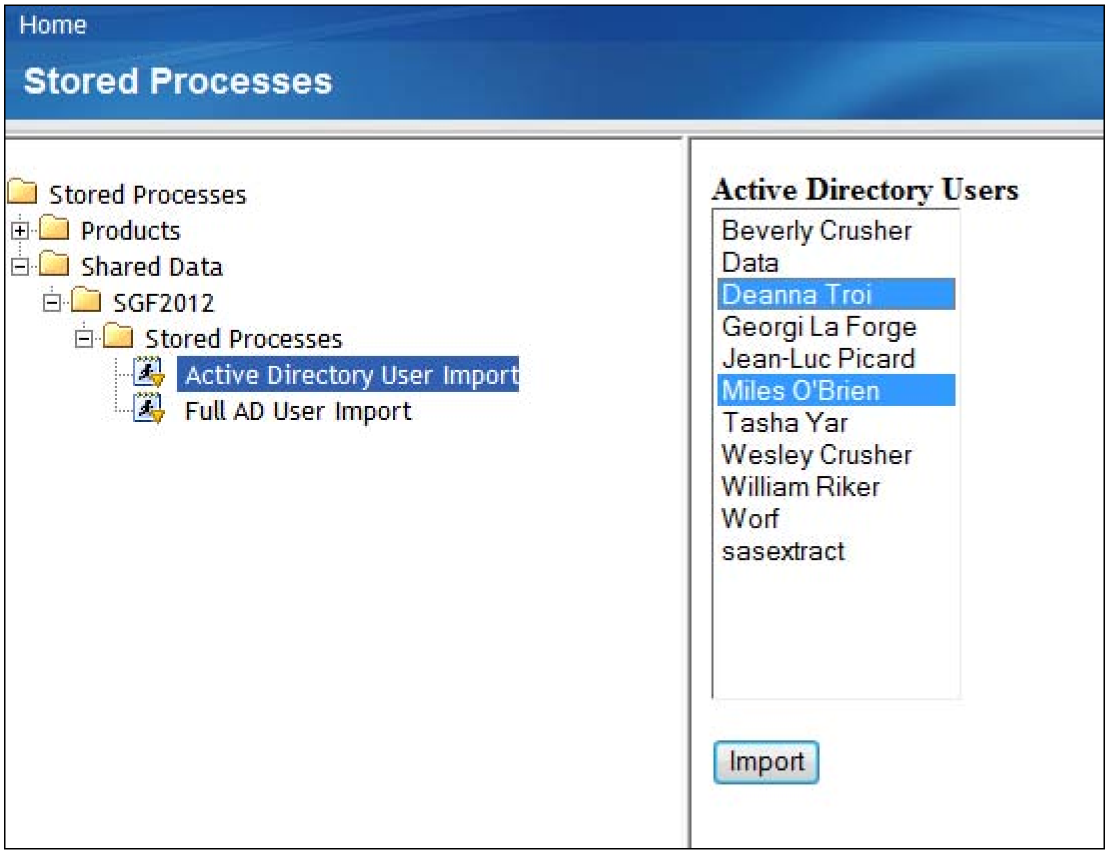


Figure 2. Custom HTML User Selection Form

First, macro variables are defined using PROC SQL to control subsequent processes.

```
proc sql;
  /** Get count of users **/
  select count(*) into :person_count from &importlibref..person;
  /** Get primary key IDs for each person **/
  select keyid into :keyid1-:keyid999999 from &importlibref..person;
  /** Get person names which match to key primary key ID **/
  select displayname into :name1-:name999999 from &importlibref..person;
quit;
```

A custom macro is written to dynamically write HTML code, which adds users to the multiple selection input form.

```
%macro person_list;
  %do i=1 %to &person_count;
    put '<option value="' "&&keyid&i" '">' "&&name&i" '</option>';
  %end;
%mend person_list;
```

This macro loops through the number of users in Active Directory and writes the “keyid” as the option value and the full name as the displayed value in the input form. It is more efficient to use the “keyid” as the option value because this value is passed into the next stored process. Using a unique identifier eliminates conflicts with redundant user names.

The following data step is used to write the custom HTML form in the stored process.

```
data _null_;
  file _webout;
  put '<html>';
  put '<form name="Users" method="post" action=' "&_THISSESSION" '>';
  put '<input type="hidden" id="_program" name="_program"
    value="/Shared Data/SGF2012/Stored Processes/Active Directory User Import 2" />';
  put '<b>Active Directory Users</b>';
  put '<br>';
  put '<select multiple="multiple" size=15 name="user">' ;
  %person_list;
  put '</select>';
  put '<br>';
  put '<br>';
  put '<input type="submit" value="Import">' ;
  put '</html>';
run;
```

The “post” action for this HTML form is to execute the stored process defined in the hidden input field with ID of “_program”. The &_THISSESSION macro variable resolves to the root URL path for the Stored Process Server, which includes the shared session created earlier. The full URL this program submits to launch the next chained stored process is the following.

http://localhost:8080/SASStoredProcess/do?_SESSIONID=A18B5F13-7ACB-47B7-9632-FAAA6F6FE5FE&_program=/Shared+Data/SGF2012/Stored+Processes/Active+Directory+User+Import+2

The session ID is generated randomly each time the stored process is run. The stored process name is automatically HTML encoded for the URL path.

The %person_list macro is executed between the HTML <select> tags to produce the user list for the form element.

The following macro variables are prepended with “SAVE_” in order for the next chained stored process to access.

```
%let SAVE_importlibref = &importlibref;
%let SAVE_ADExtIDTag = &ADExtIDTag;
```

WRITING THE USER IMPORT STORED PROCESS

The User Import stored process is triggered by the User Selection stored process. Since the User Import stored process should never execute as a standalone process, it is hidden from users by checking the Hide from user checkbox when viewing the properties in metadata, which is shown in figure 3.

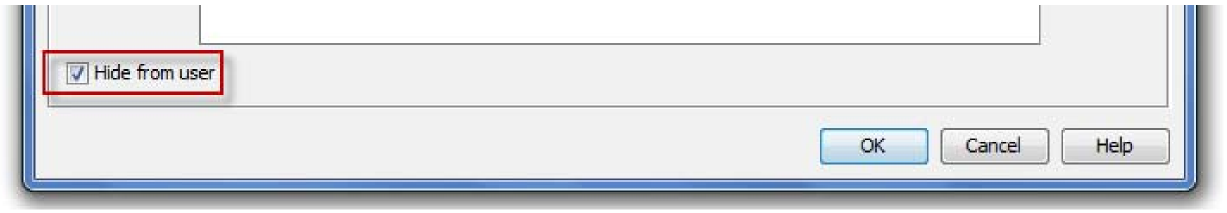


Figure 3. Hiding a Stored Process from Users

DEFINING OPTIONS FOR METADATA IMPORT

The following options are defined in the beginning of the User Import stored process to write to SAS metadata.

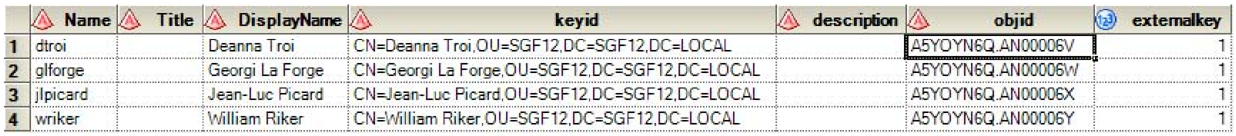
```
/* network name/address of the metadata server. */
options metaserver=localhost
/* Port Metadata Server is listening on.*/
metaport=8561
/* Domain Qualified Userid for connection to metadata server. */
metauser="sasadm@saspw"
/* Password for userid above. */
metapass="XXXXXXXXXXX"
/* Protocol for Metadata Server.  */
metaprotocol=bridge
/* Default location of user information is in the foundation repository. */
metarepository=foundation;
```

CLEANUP USER DATA PRIOR TO IMPORT

SAS datasets are created in the User Selection stored process that contain all user data from Active Directory. Before these datasets can be imported into SAS metadata they must be cleansed to prevent errors and filtered so that only select data is stored. Duplicate users will return errors. To identify these users, the current user list is generated using the following SAS macro.

```
%mduextr(libref=work);
```

This macro generates useful information about users and groups. A filtered view of the Person table is shown in Figure 4. Only users which have been created from external sources are shown. This is determined by the externalkey column equal to 1.



	Name	Title	DisplayName	keyid	description	objid	externalkey
1	dtroi		Deanna Troi	CN=Deanna Troi,OU=SGF12,DC=SGF12,DC=LOCAL		A5YOYN6Q.AN00006V	1
2	glforge		Georgi La Forge	CN=Georgi La Forge,OU=SGF12,DC=SGF12,DC=LOCAL		A5YOYN6Q.AN00006W	1
3	jlpicard		Jean-Luc Picard	CN=Jean-Luc Picard,OU=SGF12,DC=SGF12,DC=LOCAL		A5YOYN6Q.AN00006X	1
4	wriker		William Riker	CN=William Riker,OU=SGF12,DC=SGF12,DC=LOCAL		A5YOYN6Q.AN00006Y	1

Figure 4. Person Table containing users in SAS metadata

Similar to the custom HTML form, a macro must be written which produces a list of the users selected. For the user import stored process, the user list is used in PROC SQL to extract only selected users from the Active Directory extract. This macro is shown below.

```
%macro user_list;
  %if %symexist(USER_COUNT) %then %do;
    %do i=1 %to &USER0;
      "&&USER&i"
    %end;
  %end;
  %else %do;
    "&USER"
  %end;
%mend user_list;
```

The following PROC SQL statements remove users, which are already in SAS metadata, and users, which were not selected from the User Selection stored process.

```
proc sql;
  delete from &SAVE_importlibref..email
  where trim(keyid) NOT IN (%user_list);

  delete from &SAVE_importlibref..location
  where trim(keyid) NOT IN (%user_list);

  delete from &SAVE_importlibref..logins
  where trim(keyid) NOT IN (%user_list);

  delete from &SAVE_importlibref..person
  where trim(keyid) NOT IN (%user_list);

  /** Move users which are already in metadata **/
  create table excp_users as
    select * from &SAVE_importlibref..person person
    where trim(keyid) IN (
      select trim(keyid) from work.person where externalkey = 1
    );

  /** Remove users from incoming tables that are already in metadata **/
  delete from &SAVE_importlibref..person
  where trim(keyid) IN ( select keyid from excp_users );

  delete from &SAVE_importlibref..email
  where trim(keyid) IN ( select keyid from excp_users );

  delete from &SAVE_importlibref..location
  where trim(keyid) IN ( select keyid from excp_users );
  delete from &SAVE_importlibref..logins
  where trim(keyid) IN ( select keyid from excp_users );

  /** Get count of users imported **/
  select count(*) into :users_to_import from &SAVE_importlibref..person;

  /** For the scope of this paper, groups will not be imported. Remove all
  data related to groups. **/
  delete from &SAVE_importlibref..grpmems;
  delete from &SAVE_importlibref..idgrps;
quit;
```

LOAD USERS INTO SAS METADATA

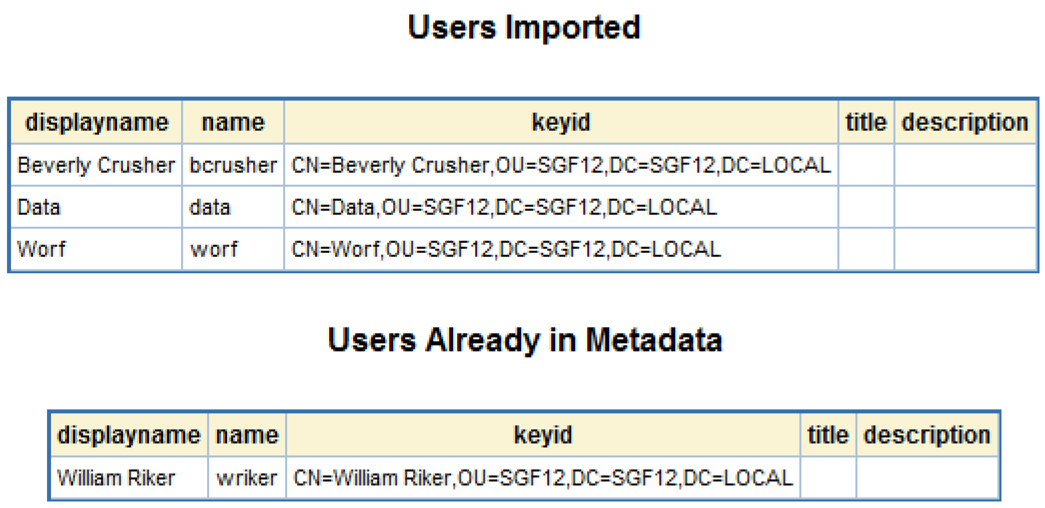
After the canonical datasets produced from the Active Directory extract are cleansed, the data can be imported into SAS metadata. The following macro was copied from the IMPORTAD.SAS program mentioned in the XXXXXXX section of this paper.

```
%macro Execute_Load;
  /* if the _EXTRACTONLY macro is set, then return and don't
  do any load processing. */
  %if %symexist(_EXTRACTONLY) %then %return;
  /* if there are no users to import, the return and do not process empty tables */
  %if &users_to_import = 0 %then %return;
  %mduimplb(libref=&SAVE_importlibref,extidtag=&SAVE_ADExtIDTag);
%mend Execute_Load;
%Execute_Load;
```

Note the SAVE_importlibref and SAVE_ADExtIDTag macro variables which were defined from the User Selection stored process.

OUTPUT USERS IMPORTED OR NOT IMPORTED

After users are imported into SAS metadata, this stored process displays the Person dataset using a PROC PRINT step. It also displays the exception table "excp_users" defined in the cleanup step in the User Import stored process. This table contains the users that were selected but already existed in SAS metadata. An example output is shown below in Figure 5.



Users Imported

displayname	name	keyid	title	description
Beverly Crusher	bcrusher	CN=Beverly Crusher,OU=SGF12,DC=SGF12,DC=LOCAL		
Data	data	CN=Data,OU=SGF12,DC=SGF12,DC=LOCAL		
Worf	worf	CN=Worf,OU=SGF12,DC=SGF12,DC=LOCAL		

Users Already in Metadata

displayname	name	keyid	title	description
William Riker	wriker	CN=William Riker,OU=SGF12,DC=SGF12,DC=LOCAL		

Figure 5. Table output showing users imported or not imported

CLOSE THE SHARED SESSION

To end the user import process, the shared stored process session is closed using the following code.

```
%let rc=%sysfunc(stpsrv_session(delete));
```

CONCLUSION

Managing a large user base in SAS is tedious and prone to human error. For IT environments that leverage Active Directory, this can be rectified by the process defined in this paper. The chained stored processes can be enhanced to import group memberships and other user information, such as job titles, emails, and descriptions. The custom HTML form can be enhanced to provide a better user interface. The demonstrated chained stored processes in this paper sets a foundation for the savvy developer to continue building.

REFERENCES

- SAS code, which demonstrates the two stored processes described in this paper is available at http://www.stephenoverton.net/SASCode/SGF2012/
- Wikipedia. “Active Directory.” March 4, 2012. Available at http://en.wikipedia.org/wiki/Active_Directory
- SAS Support. “SAS® 9.2 Intelligence Platform: Security Administration Guide - Authentication Mechanisms.” March 4, 2012. Available at http://support.sas.com/documentation/cdl/en/bisecag/61133/HTML/default/viewer.htm#a003134712.htm
- SAS Support. “SAS® 9.2 Intelligence Platform: Security Administration Guide - User Import Macros.” March 4, 2012. Available at http://support.sas.com/documentation/cdl/en/bisecag/61133/HTML/default/viewer.htm#a003175544.htm
- SAS Support. “SAS® 9.3 Stored Processes: Developer's Guide - Chaining Stored Processes.” March 4, 2012. Available at http://support.sas.com/documentation/cdl/en/stpug/62758/HTML/default/viewer.htm#datapass.htm
- SAS Support. “SAS® 9.3 Stored Processes: Developer's Guide - Using Sessions in a Sample Web Application.” March 4, 2012. Available at http://support.sas.com/documentation/cdl/en/stpug/62758/HTML/default/viewer.htm#p1m8j53px3ckx7n0zngtlmtm29ev.htm
- SAS Support. “SAS® 9.1.3 Integration Technologies Developer’s Guide - Sessions.” March 4, 2012. Available at http://support.sas.com/rnd/itech/doc9/dev_guide/stprocess/sessions.html
- Miller, Debbie. April 2005. “A Sample Macro for Creating a List on the Fly.” SUGI 30 Proceedings. Denver, CO. Available at http://www2.sas.com/proceedings/sugi30/030-30.pdf
- Wikipedia. “Lightweight Directory Access Protocol.” March 4, 2012. Available at http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol
- SAS Support. “SAS® 9.2 Intelligence Platform: Security Administration Guide - %MDUDMPLIB.” March 4, 2012. Available at http://support.sas.com/documentation/cdl/en/bisecag/61133/HTML/default/viewer.htm#a003175548.htm

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Stephen Overton
Zencos Consulting (http://www.zencos.com)

Work Email: soverton@zencos.com
Personal Email: stephen.overton@gmail.com
Personal Website: http://www.stephenoverton.net
LinkedIn: http://www.linkedin.com/in/overton