

## SAS Stored Processes

### Executing a Stored Process in SAS DMS/Batch by Interfacing with Metadata Server

Ian J. Ghent, Credit Risk Manager – Bell Canada, Ottawa, ON

#### ABSTRACT

SAS/Stored Processes are a powerful and flexible tool within the SAS BI suite. Through its standardized interface, SAS programmers, analysts, statisticians, and others, can publish reusable SAS code for other users and processes to leverage. This paper walks through the steps necessary to manually read a stored process definition in the SAS Metadata Server Repository, in order to execute the code through SAS DMS or SAS batch interfaces.

#### INTRODUCTION

A stored process is a SAS program that is stored on a server and can be executed as required by requesting applications. The location and permissions associated with the stored process are located inside the Metadata Server Repository. The original intent of stored processes was for Web reporting, analytics, building Web applications, delivering packages to clients or to the middle tier, and publishing results to channels or repositories. Stored Processes could also be very useful as change control and code standardization mechanisms. Unfortunately, there is no easy interface to the SAS DMS or batch interfaces, which many people still use. This paper will show a set of macros that can query the Metadata Server to get the location of the code used in a particular stored process.

#### METADATA SERVER

The SAS Metadata Server is the logical core of the SAS BI suite. It has a wide variety of functions including:

- User Authentication, Access Control, and Profile Provider
- Server Controller
- Systems Configuration Tracking
- Library, dataset, and OLAP Metadata storage
- Information Map Metadata Storage
- Stored Process Metadata Storage

For our purposes we need to only understand a little bit about Metadata querying and how the Stored Process Metadata is stored.

There are two main methods for querying the Metadata server:

- Standard Interface: is accessible via function/method calls in Base SAS or SAS/AF.
- DoRequest Method: is an XML based query method which wraps method calls and parameters inside custom XML tags. To use the DoRequest method, you issue a call to Proc Metadata.

Both methods return an XML stream/file which can be parsed to get at the desired information. Since the XML being returned is similar in format to the XML request of the DoRequest method, I would recommend using the DoRequest method as the default query method. This macros in this paper use the DoRequest method.

#### STORED PROCESSES

Stored Processes are stored within the Metadata Server within a tree structure, which also can house other types of information (i.e Information Maps). The location of a stored process is addressed as a path, similar to a file/directory path. A typical stored process location might be a path like this; *"/BIP Tree/Example"*. The SAS Management Console shows trees visually in the "BI Manager".

A stored process tree node is a special type of "Classifier Map", which is one of the 165 different data types that can be stored in the Metadata Server. The map stores a information needed to run a stored process, which includes:

- Which server should execute the stored process
- The location of the source code
- Input streams

- Output format (none/streaming/transient package/permanent package)
- Who has access
- Parameters which the user should be prompted to enter

#### Output

`%stpBegin` and `%stpEnd` are macros that are defined within the execution environment for the stored process. They are basically a set of ODS statements that redirect the listing and graphical output to particular ODS destinations. Each stored process which creates listing or graphical output should have that code wrapped in an `%STPBEGIN` and `%STPEND`. To execute stored processes manually, without the normal stored process execution environment, you need to create a simple definition of these macros.

#### THE MACROS

The code for the macros presented here is in the appendix. The following is a brief description of each of the macros and their function:

##### %getTreeNodeInfo(location, treeType):

- Generic macro that gets the node id and name of the node for a given path and tree type
- For stored processes the tree type is "BIP Folder"
- Returns the tree node id as a global macro variable ("treeNodeId")

##### %getStoredProcessFileId(treeNodeId):

- Gets the source file Id for a given tree node
- Returns the source file id as a global macro variable ("sourceFileId")

##### %getStoredProcessFilePath(sourceFileId):

- Gets the file path from the file id
- Returns the path as a global macro variable ("sourceFilePath")

##### %stpBegin:

- Holds ODS opening statements

##### %stpEnd:

- Holds ODS closing statements

##### %executeStoredProcess(location):

- Is the wrapper macro for the whole process
- Example of a stored process location is *"/BIP Tree/Sample"*
- Performs an `%include` on the source file path (assumes macro running on same sever)

Prior to running the macros, the metadata server options must have been submitted in the current session (i.e. `metaserver`, `metaport`, `metauser`, `metapass`, `metarepository`).

#### CONCLUSION

SAS has provided a unique and valuable way to combine custom code into BI processes. These macros allow you to leverage stored processes in every aspect of SAS usage. This paper and the associated macro code should provide a better understanding into the role that the Metadata Server plays in the SAS BI platform, and how to query it.

## APPENDIX – MACRO SOURCE CODE

```
/** This file contains macros needed to execute a stored process from the SAS
 * DMS or batch interfaces.
 * @author Ian J. Ghent<ian.ghent@bell.ca>
 * @version 1.0
 * @see #getNodeInfo()
 * @see #getStoredProcessFileId()
 * @see #getStoredProcessFilePath()
 * @see #executStoredProcess()
 */

/**
 * Gets the node id and name of the node for a given path
 * @author Ian J. Ghent<ian.ghent@bell.ca>
 * @param location path of the tree node
 * @param treeType type of tree to be searched (ie 'BIP Folder' for stored process)
 * @return treeNodeId metadata id of the tree node
 * @return treeNodeName name of the tree node (parsed from location)
 * @throws spError if an error occurred in this macro spError = 1 else 0
 */
%macro getNodeInfo(location, treeType);
 * output macro variables ;
 %GLOBAL treeNodeId;
 %GLOBAL treeNodeName;
 * error condition ;
 %GLOBAL spError;
 %LET spError = 0;

 * build up request XML ;
 filename xmlReq temp lrecl=1024;
 data _null ;
 file xmlReq;
 put '<GetMetadataObjects>';
 put ' <Reposid>$METAREPOSITORY</Reposid>';
 put ' <NS>SAS</NS>';
 put ' <Type>Tree</Type>';
 put ' <Flags>384</Flags>'; /*OMI_GET_METADATA+XMLSELECT*/
 put ' <Options>';
 put ' <XMLSelect search="*"[@TreeType=&apos;]';
 put "&treeType";
 put '&apos;]"/>';
 put ' </Options>';
 put '</GetMetadataObjects>';
run;

 * create SAS XML Map for converting tree query results into a readable dataset ;
 filename xmlMap temp lrecl=1024;
 data _null ;
 file xmlMap;
 put '<?xml version="1.0" ?>';
 put '<SXLEMAP version="1.2" name="TreeMap">';
 put ' <TABLE name="TreePaths">';
 put ' <TABLE-PATH syntax="XPath">/GetMetadataObjects/Objects/Tree</TABLE-PATH>';
 put ' <COLUMN name="treeNodeName" retain="yes">';
 put ' <PATH syntax="XPath">/GetMetadataObjects/Objects/Tree/@Name</PATH>';
 put ' <TYPE>character</TYPE>';
 put ' <DATATYPE>string</DATATYPE>';
 put ' <LENGTH>256</LENGTH>';
 put ' </COLUMN>';
 put ' <COLUMN name="treeNodeId">';
 put ' <PATH syntax="XPath">/GetMetadataObjects/Objects/Tree/@Id</PATH>';
 put ' <TYPE>character</TYPE>';
 put ' <DATATYPE>string</DATATYPE>';
 put ' <LENGTH>17</LENGTH>';
 put ' </COLUMN>';
 put ' </TABLE>';
 put '</SXLEMAP>';
run;

 * setup fileref to receive the results ;
 filename xmlResp temp lrecl=32767;
 * query the metadata server ;
 proc metadata in=xmlReq out=xmlResp;
run;
 * assign a xml libname to access the results ;
 libname treeinfo xml xmlfileref=xmlresp xmlmap=xmlmap access=readonly;

 * tokenize and parse the results to find the correct tree node ;
 %let location = %substr(&location,2);
 %let tokenCount = %sysfunc(count(&location, /));
 %let treeNodeName = %scan(&location, %eval(&tokenCount+1), /);
 %do i=1 %to &tokenCount;
 %let tokenTemp = %scan(&location, &i, /);
```

```

%let idTemp = ;
data _null_;
  set treeinfo.TreePaths;
  if treeNodeName = trim("&tokenTemp") then call symput('idTemp',treeNodeId);
run;
%if (%length(&idTemp) = 0) %then %do;
  %put %str(ERROR: SAS Stored Process Path (/&location) Not Found);
  %let spError=1;
%end;
%else %do;
  * if this is the last iteration, then output the treeNodeId of the stored process ;
  %if (&i = &tokenCount) %then %do;
    %let treeNodeId = &idTemp.;
  %end;
%end;
%end;
%if (%length(&treeNodeId) = 0) %then %do;
  %put %str(ERROR: SAS Tree Node ID Not Found);
  %let spError = 1;
%end;
libname treeinfo;
filename xmlMap;
filename xmlResp;
filename xmlReq;
%mend getTreeNodeInfo;

/**
 * Gets the sourceFileId for a given tree node
 * @author Ian J. Ghent<ian.ghent@bell.ca>
 * @param treeNodeId id of the node that contains the stored process
 * @return sourceFileId Metadata id of the source file
 * @throws spError if an error occurred in this macro spError = 1 else 0
 */
%macro getStoredProcessFileId(treeNodeId);
  * output macro variables ;
  %GLOBAL sourceFileId;
  %GLOBAL spError;
  %LET spError = 0;

  * workaround for SAS trying to resolve xml apos tags ;
  %let apos=%nrstr(&apos);

  * build up request XML to get stored process information ;
  filename xmlReq temp lrecl=1024;
  data _null_;
    file xmlReq;
    put '<GetMetadataObjects>';
    put ' <Reposid>$METAREPOSITORY</Reposid>';
    put ' <NS>SAS</NS>';
    put ' <Type>ClassifierMap</Type>';
    put ' <Flags>388</Flags>'; /*OMI_GET_METADATA+XMLSELECT+OMI_TEMPLATE*/
    put ' <Options>';
    put ' <Templates>';
    put ' <ClassifierMap>';
    put ' <SourceCode>';
    put ' <File id="" name=""/>';
    put ' </SourceCode>';
    put ' </ClassifierMap>';
    put ' </Templates>';
    put " <XMLSelect
search=""ClassifierMap[@TransformRole='StoredProcess']/Trees/Tree[@Id=&apos;&treeNodeId&apos;]""/>";
    put ' </Options>';
    put '</GetMetadataObjects>';
  run;

  * create SAS XML Map for converting stored process query results into a readable dataset ;
  filename xmlMap temp lrecl=1024;
  data _null_;
  file xmlmap;
  put '<?xml version="1.0" ?>';
  put '<SXLEMAP version="1.2" name="FileInfoMap">';
  put ' <TABLE name="FileInfo">';
  put ' <TABLE-PATH syntax="XPath">/GetMetadataObjects/Objects/ClassifierMap/SourceCode/File</TABLE-PATH>';
  put ' <COLUMN name="sourceFileId" retain="yes">';
  put ' <PATH syntax="XPath">/GetMetadataObjects/Objects/ClassifierMap/SourceCode/File/@Id</PATH>';
  put ' <TYPE>character</TYPE>';
  put ' <DATATYPE>string</DATATYPE>';
  put ' <LENGTH>17</LENGTH>';
  put ' </COLUMN>';
  put ' </TABLE>';
  put '</SXLEMAP>';
  run;

  * setup fileref to receive the results ;
  filename xmlResp temp lrecl=32767;

```

```

* query the metadata server ;
proc metadata in=xmlReq out=xmlResp;
run;
* assign a xml libname to access the results ;
libname fileinfo xml xmlfileref=xmlResp xmlmap=xmlMap access=readonly;

data _null_;
  set fileinfo.FileInfo;
  call symput('sourceFileId',sourceFileId);
run;

%if (%length(&sourceFileId) = 0) %then %do;
  %put %str(ERROR: SAS Stored Process Source FileId Not Found);
  %let spError = 1;
%end;

libname fileinfo;
filename xmlMap;
filename xmlResp;
filename xmlReq;
%mend getStoredProcessFileId;

/**
 * Gets the stored process file path a given fileId
 * @author Ian J. Ghent<ian.ghent@bell.ca>
 * @param fileId id of the node that contains source file information for the stored process
 * @return sourceFilePath fully qualified path of this stored process source file
 * @throws spError if an error occurred in this macro spError = 1 else 0
 */
%macro getStoredProcessFilePath(sourceFileId);
  * output macro variables ;
  %GLOBAL sourceFilePath;
  %GLOBAL spError;
  %LET spError = 0;

  * Get sas file information ;
  filename xmlReq temp lrecl=1024;
  data _null_;
    file xmlReq;
    put '<GetMetadata>';
    put ' <Metadata>';
    put " <File Id=""&sourceFileId"">";
    put ' <Directories>';
    put ' <Directory/>';
    put ' </Directories>';
    put " </File>";
    put ' </Metadata>';
    put ' <NS>SAS</NS>';
    put ' <Flags>276</Flags>'; /*OMI_GET_METADATA+OMI_INCLUDE_SUBTYPES+OMI_TEMPLATE*/
    put ' <Options>';
    put ' <Templates>';
    put ' <File Id="" FileName="">';
    put ' <Directories>';
    put ' <Directory/>';
    put ' </Directories>';
    put ' </File>';
    put ' <File Id="" FileName="">';
    put ' <Directory Id="" DirectoryName="">';
    put ' </Templates>';
    put ' </Options>';
    put '</GetMetadata>';
  run;

  filename xmlMap temp lrecl=1024;
  data _null_;
    file xmlMap;
    put '<?xml version="1.0" ?>';
    put '<SXLEMAP version="1.2" name="FileDirectoryMap">';
    put ' <TABLE name="FileDirectory">';
    put ' <TABLE-PATH syntax="XPath">/GetMetadata/Metadata/File</TABLE-PATH>';
    put ' <COLUMN name="filePath" retain="yes">';
    put ' <PATH syntax="XPath">/GetMetadata/Metadata/File/Directories/Directory/@DirectoryName</PATH>';
    put ' <TYPE>character</TYPE>';
    put ' <DATATYPE>string</DATATYPE>';
    put ' <LENGTH>1024</LENGTH>';
    put ' </COLUMN>';
    put ' <COLUMN name="fileName" retain="yes">';
    put ' <PATH syntax="XPath">/GetMetadata/Metadata/File/@FileName</PATH>';
    put ' <TYPE>character</TYPE>';
    put ' <DATATYPE>string</DATATYPE>';
    put ' <LENGTH>256</LENGTH>';
    put ' </COLUMN>';
    put ' </TABLE>';
    put '</SXLEMAP>';
  run;

```

```

* setup fileref to receive the results ;
filename xmlResp temp lrecl=32767;
* query the metadata server ;
proc metadata in=xmlReq out=xmlResp;
run;
* assign a xml libname to access the results ;
libname filedir xml xmlfileref=xmlResp xmlmap=xmlMap access=readonly;

%if (&sysscp = WIN) %then %let delimiter = %STR(\);
%else %let delimiter = %STR(/);

data _null_;
  set filedir.FileDirectory;
  call symput('sourceFilePath',trim(filePath) || "&delimiter" ||trim(fileName));
run;

%if (%length(&sourceFilePath) = 0) %then %do;
  %put %str(ERROR: SAS Stored Process Source FileId Not Found);
  %let spError = 1;
%end;
libname filedir;
filename xmlMap;
filename xmlResp;
filename xmlReq;
%mend getStoredProcessFilePath;

/**
* Is a filler for required macros to run stored process. This should contain ODS opening statements.
* @author Ian J. Ghent<ian.ghent@bell.ca>
*/
%macro stpBegin();
%mend stpBegin;

/**
* Is a filler for required macros to run stored process. This should contain ODS closing statements.
* @author Ian J. Ghent<ian.ghent@bell.ca>
*/
%macro stpEnd();
%mend stpEnd;

/**
* Acts as a wrapper executing 3 other macros to get all the information to run a stored process
* @author Ian J. Ghent<ian.ghent@bell.ca>
* @param location path of this tree node
* @return sourceFilePath fully qualified path of this stored process source file
* @throws spError if an error occurred in this macro spError = 1 else 0
* @see #getNodeInfo()
* @see #getStoredProcessFileId()
* @see #getStoredProcessFilePath()
*/
%macro executeStoredProcess(location);
  %GLOBAL spError;
  %LET spError = 0;

  %getNodeInfo(&location, BIP Folder);

  %if (&spError < 1) %then %do;
    %getStoredProcessFileId(&treeNodeId);
  %end;
  %if (&spError < 1) %then %do;
    %getStoredProcessFilePath(&sourceFileId);
  %end;
  %if (&spError < 1) %then %do;
    %put %str(***** START STORED PROCESS: &treeNodeName *****);
    %include "&sourceFilePath";
    %put %str(***** END STORED PROCESS *****);
  %end;
  %else %put %str(ERROR: Stored Process failed to execute);

  * cleanup global macro variables ;
  %syndel treeNodeId treeNodeName sourceFileId sourceFilePath;
  run;
%mend executeStoredProcess;

* example execution ;
*options metaserver=localhost;
*options metaport=8561;
*options metauser="";
*options metapass="";
*options metarepository=Foundation;
*%executeStoredProcess(/BIP Tree/Hours Worked By Department);

```