Paper 061-2011

Impact Analysis – The Smart Way

George Mendoza, CIGNA Healthcare, Bloomfield, CT

ABSTRACT

Very often, Business Analysts and SAS® programmers need to analyze the impacts of making changes to existing tables, datasets, macro variables etc. In order to fully understand the impact of the changes it is critical to list all SAS codes that use or reference the objects (tables, datasets, macro variables etc.) that are changing. This paper describes an innovative method for listing all the impacted code.

INTRODUCTION

There are many methods to search for impacts of making changes to existing tables, datasets and macro variables. For example, on Windows we can use the windows search functionality and on Unix we can use Perl scripts. However, these methods can be time-consuming when there are multiple folders to search and in the case of perl scripts, the scripting language syntax can also be challenging. What we need is a simple process to allow both developers and business analysts to easily list the elements that are impacted.

THE CONCEPT

When we search for a particular element being referenced by code, we are actually searching for a string/word in the code file. What we need to do is to create a SAS dataset which lists every word from all code files in all code directories together with the full path to the code. This dataset can then be used to query a string value and thereby list all the codes that reference a particular object.

For example, consider Company XYZ has 100 code files in each of 10 code directories. Some of this code references dataset dsname. The dataset dsname is going to be replaced and the codes that use this dataset now need to use a new dataset. Hence, we'll need to review all 100*10 code files to determine which code is impacted. If we had a SAS dataset that contained all words from all code files together with the path to the code, we could run a query for the word "dsname" to list all code files that use this dataset. That is all there is to understand! The rest of the paper will describe how to code the logic to accomplish this.

THE CODE

We need to write code to accomplish the following

1. List all code files in the user specified directories
2. Read each code file from each input directory and write all words to an output dataset.

The following sections will explain parts of the code.

1. LIST ALL CODE FILES IN DIRECTORY:

The PIPE device on a FILENAME statement allows you to specify a command, such as the dir command on Windows, or the ls command on Unix. A subsequent DATA step with an INFILE statement that references the fileref created by the FILENAME statement can then read the stream produced by the dir or ls commands.

Figure 1.
Code to set the pipe command based on the operating system

```
%if "&SYSSCP" = "WIN" %then
%do;
   filename dirlist pipe "dir &dirPath /b";
%end;
%else
 %do;
  filename dirlist pipe "ls &dirPath";
 %end;
```

Begin by checking the operating system because the commands on Unix and Windows are different. The operating system is available from the &SYSSCP macro variable. In Figure 1, &SYSSCP and &dirPath are macro variables which have the values of the operating system (WIN, AIX etc.) and directory path respectively.

Figure 2.
Code to create a list of all files in a directory

```
data dirFiles(keep=dirName fileName);
infile dirlist truncover;
length dirName $100;
input fileName $char100.;
dirName = "&dirPath";
if trim("&fileFilter") ne '' then
 do;
 if index(fileName,trim("&fileFilter")) > 0
 then output;/*Add specified file filter*/
 end;
else output;
run;
```

Create a dataset containing all the files within the directory using the pipe command from Figure 1 above. In the next step, we will loop through this dataset and retrieve all words from each of these code files. In this example, the &fileFilter macro variable is set to ".sas" so that we only process SAS code files.

2. READ ALL WORDS FROM A FILE AND WRITE THEM TO A SAS DATASET

Figure 3.
Read words from a file

```
data wordFile (keep = fileFullPath word);
infile "&fileFullPath";
length word $250;
input word $ @@;
```

Notice how the @@ will ensure that we read one word at a time. The &fileFullPath macro variable is set to the full path of the code file.

Figure 4.
Strip unwanted characters

```
word = compress(wordRead, '09'x);
```

You may want to strip out unwanted characters from the word. In Figure 4 we are stripping out the tab character.

Figure 5.
Do not write commented text to output file

```
if index(word,'/*') > 0 and commentFlag = 'N'
then
do;
 word=scan(word,1,'/*');
 if trim(word) ne '' then output;
 commentFlag = 'Y';
end;

if index(word,'*/') > 0 and commentFlag = 'Y'
then
do;
 word=scan(word,-1,'*/');
 if trim(word) ne '' then output;
 commentFlag = 'N';
end;
```

You may also want to ignore the text in comments. To do this, look for "/*" appearing in the word. If found, stop writing to the file until the closing comment "*/" is found.

USAGE

Figure 6.
Use the results to complete your impact analysis!

```
Proc sql;
Create table impactedCode as
Select *
From allWords
Where upcase(word) like ‘ %DSNAME%’;
Quit;
```

Now that all words are captured in the “allWords” dataset we can query it to determine which codes use the “dsname” dataset that is changing. We can use a query similar to the one shown in Figure 6.

THE COMPLETE CODE

```
/*==============================================================================*/
/* Input parameters
/* 1. numberOfDirectories = Total number of input libraries
/* 2. dirPathn = Directory path n. Where n=1,2,3, etc.
/* 3. fileFilter = File type to process. For sas code files specify .sas
/* 4. charsToOmit = Characters to be stripped from the words before
/*                  writing to the output file.
/*==============================================================================*/
%let numberOfDirectories = 2;
%let dirPath1 = /folder1/subfolder1/subfolder2/;
%let dirPath2 = /folder2/subfolder1/subfolder2/;
%let fileFilter = .sas;
%let charsToOmit = '?();,-=<>';
run;

/*===============================================================================*/
/* Do not change anything below this line
/*===============================================================================*/

%macro listFilesInDir(dirPath, env, fileFilter);
/* PURPOSE   : Create listing of files in specified directory on Unix or Windows
/* PARAMETERS: 1. dirPath - this is the full path of the directory
/*             2. env - this is the operating system we are using Unix/Windows
/*             3. fileFilter - we only want to return those files with a particular
/*                            extension. e.g. .sas. Leave blank if not required
*/

/*===============================================================================*/
/* Assign appropriate file listing pipe command for Unix or Windows environment
/*===============================================================================*/
      %if "&SYSSCP" = "WIN" %then
        %do;
           filename dirlist pipe "dir &dirPath /b";
        %end;
      %else
        %do;
           filename dirlist pipe "ls &dirPath";
        %end;
```

```
/*===============================================================================*/
/* Create dataset with all files in directory
/*===============================================================================*/
      data dirFiles(keep=dirName fileName);
           infile dirlist truncover;
           length dirName $100;
           input fileName $char100.;
      dirName = "&dirPath";
      if trim("&fileFilter") ne '' then
         do;
         /*Add specified file filter*/
         if index(fileName,trim("&fileFilter")) > 0 then output;
         end;
      else output;
  run;

%mend lisFilesInDir;

%macro readWordsFromFile(fileFullPath,charsToOmit);
/* PURPOSE    : Create a dataset containing all the words in an input file
/* PARAMETERS: 1. fileFullPath - this is the full path to the file
/*             2. charsToOmit - these are the characters to be stripped from the
/*                                words e.g. ;"",/ etc.
*/

   data wordFile (keep = fileFullPath word);
        infile "&fileFullPath";
        length word wordRead $250 fileFullPath $200;
        input wordRead $ @@;
        retain commentFlag 'N';

 /*==============================================================================*/
 /* Remove unwanted characters from the word that was read
 /*==============================================================================*/
   wordRead = compress(wordRead,"&charsToOmit" || '09'x);
   fileFullPath = "&fileFullPath";

 /*===============================================================================*/
 /* Check for start of SAS comment. We don't want to output commented text
 /*===============================================================================*/
   if index(wordRead,'/*') > 0 and commentFlag = 'N' then
      do;
         word=scan(wordRead,1,'/*');
         if trim(word) ne '' then output;
         commentFlag = 'Y';
      end;

/*===============================================================================*/
/* Check for end of SAS comment. We want to output text after the comment is
/* complete
/*==============================================================================*/
  if index(wordRead,'*/') > 0 and commentFlag = 'Y' then
     do;
       word=scan(wordRead,-1,'*/');
       if trim(word) ne '' then output;
       commentFlag = 'N';
     end;
```

```
/*============================================================================*/
/* Write word to output after verifying that it is not commented
/*=============================================================================*/
  else if commentFlag = 'N' and trim(wordRead) ne '' then
       do;
        word= wordRead;
        output;
         end;
  run;
%mend readWordsFromFile;

%macro createWordDataset(fileFullPath,charsToOmit);
/* PURPOSE   : Retrieve words from a file and add the words to a SAS dataset
/* PARAMETERS: 1. fileFullPath - this is the full path to the file
/*             2. charsToOmit - these are the characters to be stripped from the
/*                              words e.g. "",/ etc.
/*============================================================================*/
/* Run macro to produce word lost from file
/*============================================================================*/
   %readWordsFromFile(&fileFullPath,&charsToOmit);

/*===========================================================================*/
/* Append words from current file to dataset containing words from all code files
/*============================================================================*/
   proc append base=allWords data=wordFile;run;

/*============================================================================*/
/* Delete temporary file that has the words for current file
/*=============================================================================*/
   proc datasets library=work;delete wordFile;run;
%mend createWordDataset;

%macro runProcess;
/* PURPOSE   : This is the main macro which runs all other macros to produce as
/*             SAS dataset containing all words used in code files present in the
/*             folders specified by the user
*/
/*============================================================================*/
/* Create dataset that stores filenames from all input folders
/*============================================================================*/
  proc sql;
   create table allSASCodes
                (dirName char(100),
                 fileName char(100));
  quit;

/*=============================================================================*/
/* Retrieve all code files in input folders and save the file names in one dataset
/*============================================================================*/
  %do i=1 %to &numberOfDirectories;
      %listFilesInDir(&&dirPath&i,&env,&fileFilter);
      proc append base=allSASCodes data=dirFiles;run;
      proc datasets library=work;delete dirFiles;run;
  %end;
/*==========================================================================*/
/* Create dataset that stores words from all files
/*==========================================================================*/
 proc sql;
  create table allWords
      (fileFullPath char(200),
       word char(250));
 quit;
```

```
/*==============================================================================*/
/* Retrieve Words from all files
/*===============================================================================*/
data _null_;
set allSASCodes;
parm = '%createWordDataset(' || trim(dirName) || trim(fileName) ||
                                   ','|| "&charsToOmit" || ')';
call execute(parm);
run;

%mend;
/*===============================================================================*/
/* run the main macro program to retrieve the names of all code files in user
/* specified code directories and create a dataset containing all words used in
/* these code files
/*==============================================================================*/
%runProcess;
```

CONCLUSION

Thus, by creating a SAS dataset that contains the words used by code files in all code directories, we are able to simplify the daunting task of searching for impacts. This process can also be used to get a better understanding of how tables, datasets and variables are being used by the organization.

REFERENCES

- Ronald Cody, Ed.D.. 2007. “An Introduction to SAS® Character Functions.” Proceedings of the SAS Global Forum 2007 Conference. Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

Thanks to the SAS online support for prompt responses to my questions.

Many thanks also to Cheryl Harmon, Jennifer Veilleux, Tanya Iwanski and Michael McDermott for their positive feedback when I introduced this concept in my project.

A special thank you to Mike Rhoads for his excellent review and suggestions.

DISCLAIMER

CONTACT INFORMATION

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

George Mendoza
CIGNA Healthcare
900 Cottage Grove Road
Bloomfield, CT 06152
Work Phone: 860-226-1960
Email: mendoza_george@yahoo.com

* *