**Producing code diagrams using “Enhanced Analysis”**

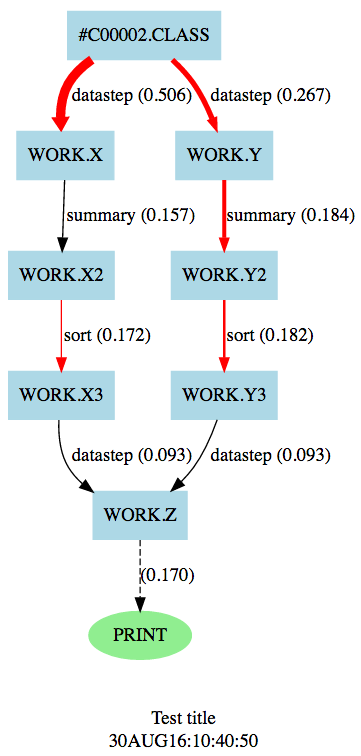
**Introduction**

PROC SCAPROC has been available in SAS since SAS 9.2. It implements the SAS Code Analyzer which provides information about input tables, output tables and macro symbols in a SAS program. My “Enhanced Analysis” uses the data generated by PROC SCAPROC to generate DOT directives which can then be used to generate a diagram of the flow of the program DOT is a plain text graph description language. For instance specifying “A-B” will draw 2 circles A and B connected by a line.

**To use this on your own SAS program**:

1. You need to have the macros accessible either via an autocall library or just run them first.
2. Put the %EANBEGIN macro before your code.
3. Put the %EANEND macro after your code.
4. Run those macro with your code.
5. Run SCAPROC\_ANALYSE
6. Copy the lines from the table just created (GRAPHVIZ).
   1. View the table in EG.
   2. Click on column, to select all values in it
   3. Then control-C.
7. Go to [http://webgraphviz.com](http://webgraphviz.com/)
   1. Paste the lines into the Text Area.
   2. Press "Generate Graph!" button.
8. Now you will have your diagram displayed in your web browser.
   1. You can copy it or you could save it as:
      1. SVG (small and scalable, you can scale it as big or small as you want to without losing quality)
      2. PNG (small and not scalable)
      3. BMP (big and not scalable)

**Sample Output**



**DOT code that was produced**

// Generated by SAS for Test title

// Percentiles: 50:0.172 60:0.182 70:0.184 80:0.267 90:0.506 95:0.506 99:0.506

digraph test {

graph [label="\n\nTest title\n30AUG16:10:40:50"]

node [shape=box color=lightblue style=filled]

"PRINT"[shape=ellipse color=lightgreen]

"#C00002.CLASS"->"WORK.X" [label=" datastep (0.506)" color=red penwidth=7 style=solid];

"#C00002.CLASS"->"WORK.Y" [label=" datastep (0.267)" color=red penwidth=4 style=solid];

"WORK.X"->"WORK.X2" [label=" summary (0.157)"   style=solid];

"WORK.Y"->"WORK.Y2" [label=" summary (0.184)" color=red penwidth=3 style=solid];

"WORK.X2"->"WORK.X3" [label=" sort (0.172)" color=red  style=solid];

"WORK.Y2"->"WORK.Y3" [label=" sort (0.182)" color=red penwidth=2 style=solid];

"WORK.X3"->"WORK.Z" [label=" datastep (0.093)"   style=solid];

"WORK.Y3"->"WORK.Z" [label=" datastep (0.093)"   style=solid];

"WORK.Z"->"PRINT" [label="(0.170)"   style=dashed];

}

**SAS Code that does an “Enhanced Analysis”**

%let \_eandebug=scaproc ;

%***eanbegin***(Sample **1**)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* This is the sample program we will measure and then make a flow chart ;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**data** x ;

   set sashelp.class ;

**run** ;

**data** y ;

   set sashelp.class ;

**run** ;

**proc** **summary** data=x ;

   class sex ;

   var height ;

   output out=x2 mean= ;

**run** ;

**proc** **summary** data=y ;

   class sex ;

   var height ;

   output out=y2 mean= ;

**run** ;

**proc** **sort** data=x2 out=x3 ;

   by sex ;

**run** ;

**proc** **sort** data=y2 out=y3 ;

   by sex ;

**run** ;

**data** z ;

   merge x3 y3 ;

      by sex ;

**run** ;

**proc** **print** ;

**run** ;

**proc** **sql** ;

   create table sql\_table as

   select \*

   from x

   left join

   y

   on x.sex=y.sex ;

**quit** ;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\*\*\* finish of sample program ;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

%***eanend***

\* Have a look at the suggestions of how to grid enable this code – not required ;

filename cmd pipe "cat ""%sysget(HOME)/scaproc.txt""" ;

**data** \_null\_ ;

   infile cmd ;

   input ;

   put \_infile\_ ;

**run** ;

**Future Enhancements**

There are an almost unlimited number of enhancements that can be made using these basic techniques outlined here.

1. Install Graphviz in a place accessible from SAS server so that we can then use it directly from a SAS program. This will enable the graphics to be produced automatically thereby removing any manual steps.
2. Integrate output from SAS log so that we can choose to display various numbers on arrows and scale and colour them accordingly, such as Elapsed time, CPU time, records in/out, etc.
3. Fix “#” references from PROC SCAPROC with info from code and/or log to show what library they actually come from.
4. Integrate tool into a Stored Process, to allow running an entire analysis by filling in some parameters and browsing to your code.
5. Make simple version of diagram based only on code.
6. Make diagram that can be generated from a SAS log, enabling us to create diagrams on logs run in the past.
7. Create framework to store graph DOT commands and then enable comparison with others previously stored, so that we could compare performance between multiple runs, highlight changes in the structure of complex code diagrammatically, etc.
8. Provide parameters to allow changing of line attributes, box attributes, etc.
9. Add support for showing diagrams in a web page, which then will enable use of tooltips and links which will enable a lot more functionality such as popping up a box showing code used in a step directly from diagram.
10. Enhance code to use the grid analysis capabilities. This will suggest how to grid enable your SAS code, and how much time could be saved.

**Useful Links**

* PROC SCAPROC documentation - <http://support.sas.com/documentation/cdl/en/proc/68954/HTML/default/viewer.htm#n0pbvhhaw4f7f7n1ukrvacsgulx1.htm>
* Documentation for GraphViz, which enables Graph Vizualisation via directives provided in text form - <http://www.graphviz.org/>
* DOT language info & links - <https://en.wikipedia.org/wiki/DOT_(graph_description_language)>
* Web Version of GraphViz, which enables us to use GraphViz without installing it - <http://webgraphviz.com/>

**Author Contact Information**

Philip Mason, Wood Street Consultants Ltd.

Email: [phil@woodstreet.org.uk](mailto:phil@woodstreet.org.uk)

Mobile: 07894 239671

**Appendix**

**SAS Macros**

The following SAS macros are the version of these at the time of creating this document. They show the technique although the code is currently under development so it might not handle the analysis of all kinds of SAS code, and won’t have features currently being added in development. Feel free to make your own enhancements and share the code with others.

* EANBEGIN – “Enhanced Analysis” begin, which is used before some SAS code that you want to analyse. This follows the SAS naming scheme along the lines of STPBEGIN/STPEND.
* EANEND – “Enhanced Analysis” end, which is used at the end of SAS code which is being analysed. This causes the analysis to be written out and is then ready to be analysed.
* PERCENTILES – Creates a range of global macro variables in the form PCTn, where n is the percentile. E.g. PCT50 is the 50th percentile. There is a default list of them created although you can override which ones are created. e.g. *%percentiles(sashelp.air,air,pctlpts=25 50 75)*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Macro:           EANBEGIN                                                   \*

\*                                                                             \*

\* Author:          Philip Mason                                               \*

\*                                                                             \*

\* Date Created:    01/08/2016                                                 \*

\*                                                                             \*

\* Description:     Enhanced Analysis - begins enhanced analysis               \*

\*                  Macro variable \_EANDEBUG defines what kind of enhanced     \*

\*                  analysis is carried out according to the following values: \*

\*                  - scaproc = uses Proc SCAPROC                              \*

\*                  - verbose = turns on extra verbose logging                 \*

\*                  values for \_EANDEBUG should be comma separated             \*

\*                                                                             \*

\* Parameters:      use\_label = optionally specify a label to use in .DOT      \*

\*                              output that is produced.                       \*

\*                  where = optionally specify a location for PROC SCAPROC     \*

\*                          to write its output to, will default to a text     \*

\*                          file in your own home directory                    \*

\*                  \_options = optionally specify options to use when using    \*

\*                             PROC SCAPROC                                    \*

\*                                                                             \*

\* Data Sources:    n/a                                                        \*

\*                                                                             \*

\* Data Output:     &where defines where to save SCAPROC output                \*

\*                                                                             \*

\* Auxiliary Files: n/a                                                        \*

\*                                                                             \*

\*-----------------------------------------------------------------------------\*

\* Modification History                                                        \*

\* Date        By   Details                                                    \*

\* 01/08/2016  PM   Original Coding                                            \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**%macro** eanbegin(use\_label,

                where=%sysget(HOME)/scaproc.txt,

                \_options=attr opentimes expandmacros) ;

   %\* if \_eandebug macro exists and is not set to 0 or off, then continue ;

   %if %symexist(\_eandebug)=**0** %then %return ;

   %if &\_eandebug=**0** or %upcase(&\_eandebug)=OFF %then %return ;

   %if %superq(use\_label)= %then %let use\_label=Program run by &sysuserid ;

   %\* process parameters ;

   %let n\_parms=%eval(%sysfunc(count(%superq(\_eandebug),%str(,)))+1) ;

   %do i=**1** %to &n\_parms ;

      %let parm=%upcase(%scan(%superq(\_eandebug),&i,%str(,))) ;

      %put INFO: EANBEGIN macro invoked: &parm ;

      %if &parm=SCAPROC %then %do ;

         \* Start recording info to a text file ;

         proc scaproc ;

            record "&where" &\_options ;

         run ;

         %global scaproc\_label ;

         %let scaproc\_label=&use\_label ;

         %end ; /\* scaproc \*/

      %else %if &parm=VERBOSE %then %do ;

         \* Turn on various extra logging ;

         options symbolgen

                 mlogic mlogicnest

                 mprint mprintnest

                 mautolocdisplay mautocomploc

                 msglevel=i ;

         %\* look at macro variables before process ;

         %put \_all\_ ;

         %end ; /\* verbose \*/

      %end ; /\* do \*/

**%mend** eanbegin ;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Macro:           EANEND                                                     \*

\*                                                                             \*

\* Author:          Philip Mason                                               \*

\*                                                                             \*

\* Date Created:    01/08/2016                                                 \*

\*                                                                             \*

\* Description:     Enhanced Analysis - ends enhanced analysis                 \*

\*                                                                             \*

\* Parameters:      n/a                                                        \*

\*                                                                             \*

\* Data Sources:    n/a                                                        \*

\*                                                                             \*

\* Data Output:     as defined by the invocation of macro EANBEGIN             \*

\*                                                                             \*

\* Auxiliary Files: n/a                                                        \*

\*                                                                             \*

\*-----------------------------------------------------------------------------\*

\* Modification History                                                        \*

\* Date        By   Details                                                    \*

\* 01/08/2016  PM   Original Coding                                            \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**%macro** ***eanend*** ;

   %\* if \_eandebug macro exists and is not set to 0 or off, then continue ;

   %if %symexist(\_eandebug)=**0** %then %return ;

   %if &\_eandebug=**0** or %upcase(&\_eandebug)=OFF %then %return ;

   %\* process parameters ;

   %let n\_parms=%eval(%sysfunc(count(%superq(\_eandebug),%str(,)))+1) ;

   %do i=**1** %to &n\_parms ;

      %let parm=%upcase(%scan(%superq(\_eandebug),&i,%str(,))) ;

      %put INFO: EANBEGIN macro invoked: &parm ;

      %if &parm=SCAPROC %then %do ;

         \* write out the recorded info ;

         proc scaproc ;

            write ;

         run ;

         %end ; /\* scaproc \*/

      %else %if &parm=VERBOSE %then %do ;

         %\* look at macro variables after process ;

         %put \_all\_ ;

         %end ; /\* verbose \*/

      %end ; /\* do \*/

**%mend** eanend ;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Macro:           PERCENTILES                                                \*

\*                                                                             \*

\* Author:          Philip Mason                                               \*

\*                                                                             \*

\* Date Created:    01/08/2016                                                 \*

\*                                                                             \*

\* Description:     Generate percentiles using a variable from a table and     \*

\*                  then write them into a series of macro variables           \*

\*                                                                             \*

\* Parameters:      dset = dataset to use                                      \*

\*                  var = variable to use from dataset for generating %tiles   \*

\*                  pctlpts = list of percentiles to generate, separated by    \*

\*                            spaces. A global macro variable will be created  \*

\*                            for each one. e.g. 10 would produce &pct10       \*

\*                                                                             \*

\* Data Sources:    &dset                                                      \*

\*                                                                             \*

\* Data Output:     \_stats                                                     \*

\*                                                                             \*

\* Auxiliary Files: n/a                                                        \*

\*                                                                             \*

\*-----------------------------------------------------------------------------\*

\* Modification History                                                        \*

\* Date        By   Details                                                    \*

\* 01/08/2016  PM   Original Coding                                            \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**%macro** percentiles(dset,var,pctlpts=**50** **60** **70** **80** **90** **95** **99**) ;

   \* work out some percentiles ;

   proc univariate data=&dset noprint ;

      var &var ;

      output out=\_stats

             pctlpts  = &pctlpts

             pctlpre = pc

             pctlname =

             %do i=**1** %to %eval(%sysfunc(count(&pctlpts,%str( )))+**1**) ;

                %let item=%scan(%superq(pctlpts),&i,%str( )) ;

                t&item

                %end ;

             ;

   run ;

   \* put the percentiles into global macro variables ;

   data \_null\_ ;

      length varname $ **32** ;

      dsid=open("\_stats") ;

      call set(dsid) ;

      rc=fetch(dsid) ;

      do i=**1** to attrn(dsid,'nvars') ;

         varname=varname(dsid,i) ;

         value=getvarn(dsid,i) ;

         call execute('%global '||varname||';') ;

         call symputx(varname,value) ;

      end ;

      dsid=close(dsid) ;

   run ;

**%mend** percentiles ;

**SAS Program**

There is only one SAS program currently required.

* SCAPROC\_ANALYSE – Reads in a text file containing the output from PROC SCAPROC and produces a table containing DOT directives. These are a set of directives used with a program called GraphViz which enable graphs & diagrams to be produced. There are some manual steps that must then be taken which are outlined in the description below.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Program:         SCAPROC\_ANALYSE                                            \*

\*                                                                             \*

\* Author:          Philip Mason                                               \*

\*                                                                             \*

\* Date Created:    01/08/2016                                                 \*

\*                                                                             \*

\* Description:     Analyse output from PROC SCAPROC, which has previously     \*

\*                  been written to a text file.                               \*

\*                  After running this ...                                     \*

\*                  Now there are some manual steps. These could be automated, \*

\*                  but would need to install some software or use the Stored  \*

\*                  Process Web App.                                           \*

\*                  1 - Copy the lines from the table just created (GRAPHVIZ). \*

\*                      View the table in EG.                                  \*

\*                      Click on column, to select all values in it            \*

\*                      Then control-C.                                        \*

\*                  2 - Go to <http://webgraphviz.com/> .                        \*

\*                  3 - Paste the lines into the Text Area.                    \*

\*                  4 - Press "Generate Graph!" button.                        \*

\*                                                                             \*

\* Parameters:      n/a                                                        \*

\*                                                                             \*

\* Data Sources:    %sysget(HOME)/scaproc.txt                                  \*

\*                                                                             \*

\* Data Output:     WORK.GRAPHVIZ                                              \*

\*                                                                             \*

\* Auxiliary Files: n/a                                                        \*

\*                                                                             \*

\*-----------------------------------------------------------------------------\*

\* Modification History                                                        \*

\* Date        By   Details                                                    \*

\* 01/08/2016  PM   Original Coding                                            \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* You need to have already produced a file to analyse by using the        \*\*\*

\*\*\* eanbegin and eanend macros put around the code you want to analyse.     \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

\* list the directory to see the file created ;

filename cmd pipe "cd %sysget(HOME);ls -l;cat scaproc.txt" ;

**data** \_null\_ ;

   infile cmd ;

   input ;

   put \_infile\_ ;

**run** ;

\* read in the info and parse into a SAS table ;

filename scaproc "%sysget(HOME)/scaproc.txt" ;

**data** scaproc ;

   length word1-word6 $ **32** ;

   retain step **1** ;

   infile scaproc ;

   input ;

   put \_infile\_ ;

   if \_infile\_=:'/\* JOBSPLIT: ' ;

   word1=scan(\_infile\_,**2**,' ') ;

   word2=scan(\_infile\_,**3**,' ') ;

   word3=scan(\_infile\_,**4**,' ') ;

   word4=scan(\_infile\_,**5**,' ') ;

   word5=scan(\_infile\_,**6**,' ') ;

   word6=scan(\_infile\_,**7**,' ') ;

   if word2='DATASET' & word3='INPUT' then in=strip(word4)||'~'||scan(word5,**1**,'.')||'.'||scan(word5,**2**,'.') ;

   if word2='DATASET' & word3='OUTPUT' then out=strip(word4)||'~'||scan(word5,**1**,'.')||'.'||scan(word5,**2**,'.') ;

   if word2='DATASET' & word3='UPDATE' then out=strip(word4)||'~'||scan(word5,**1**,'.')||'.'||scan(word5,**2**,'.') ;

   if word2='PROCNAME' then procname=word3 ;

   if word2='ELAPSED' then elapsed=input(word3,**8.3**) ;

   output ;

   if index(\_infile\_,'STEP SOURCE FOLLOWS') then step+**1** ;

**run** ;

\* merge the data into one record for each step ;

**proc** **sql** noprint ;

create table flow as

   select coalesce(a.step,b.step,c.step) as step

         ,a.procname

         ,coalesce(scan(b.in,**1**,'~'),scan(c.in,**1**,'~')) as in\_access

         ,coalesce(scan(b.out,**1**,'~'),scan(c.out,**1**,'~')) as out\_access

         ,coalesce(scan(b.in,**2**,'~'),scan(c.in,**2**,'~')) as in

         ,coalesce(scan(b.out,**2**,'~'),scan(c.out,**2**,'~')) as out

         ,d.elapsed

      from

         scaproc(where=(procname>'')) as a

      full join

         scaproc(where=(in>'')) as b

         on a.step=b.step

      full join

         scaproc(where=(out>'')) as c

         on a.step=c.step

      left join

         scaproc(where=(elapsed>**0**)) as d

         on a.step=d.step

      order by calculated step

      ;

   create table procnames as

      select distinct procname

      from flow

      where procname is not missing and (missing(in) or missing(out)) ;

**quit** ;

%\* create percentiles for use in making diagram ;

%***percentiles***(flow,elapsed)

%put \_user\_ ;

\* Create .DOT directives to make a diagram ;

**data** graphviz(keep=line) ;

   length line $ **160**

         p $ **32**

         color penwidth $ **12** ;

   if \_n\_=**1** then do ;

      line="// Generated by SAS for %superq(scaproc\_label)" ;

      output ;

      line="// Percentiles: 50:&pct50 60:&pct60 70:&pct70 80:&pct80 90:&pct90 95:&pct95 99:&pct99" ;

      output ;

      line='digraph test {' ;

      output ;

/\*      line='rankdir=LR' ;\*/

/\*      output ;\*/

      line="graph [label=""\n\n%superq(scaproc\_label)\n%sysfunc(datetime(),datetime.)""]" ;

      output ;

      line='node [shape=box color=lightblue style=filled]' ;

      output ;

      dsid=open('procnames') ;

      do while(fetch(dsid)=**0**) ;

         p=getvarc(dsid,**1**) ;

         line=quote(strip(p))||'[shape=ellipse color=lightgreen]' ;

         output ;

      end ;

      dsid=close(dsid) ;

      end ;

   set flow end=end ;

   in=quote(strip(in)) ;

   out=quote(strip(out)) ;

  procname=quote(strip(procname)) ;

   if elapsed>=&pct50 then color='color=red' ;

                      else color='' ;

   if elapsed>=&pct99 then penwidth='penwidth=7' ; else

   if elapsed>=&pct95 then penwidth='penwidth=6' ; else

   if elapsed>=&pct90 then penwidth='penwidth=5' ; else

   if elapsed>=&pct80 then penwidth='penwidth=4' ; else

   if elapsed>=&pct70 then penwidth='penwidth=3' ; else

   if elapsed>=&pct60 then penwidth='penwidth=2' ;

                      else penwidth='' ;

   if in\_access='MULTI' or out\_access='MULTI' then style='style=dashed' ;

                                              else style='style=solid ' ;

   if compress(in,'"')>'' & compress(out,'"')>'' then

                               line=strip(in)||'->'||strip(out)||

                               ' [label=" '||lowcase(strip(dequote(procname)))||

                               ' ('||strip(put(elapsed,**8.3**))||

                               ')" '||strip(color)||' '||strip(penwidth)||' '||strip(style)||'];' ;

   else if compress(in,'"')>'' & compress(out,'"')='' then

                               line=strip(in)||'->'||strip(procname)||

                               ' [label="('||strip(put(elapsed,**8.3**))||

                               ')" '||strip(color)||' '||strip(penwidth)||' '||strip(style)||'];' ;

   else if compress(in,'"')='' & compress(out,'"')>'' then

                               line=strip(procname)||'->'||strip(out)||

                               ' [label="('||strip(put(elapsed,**8.3**))||

                               ')" '||strip(color)||' '||strip(penwidth)||' '||strip(style)||'];' ;

   else line='// '||strip(procname)||' ('||strip(put(elapsed,**8.3**))||')' ;

   output ;

   if end then do ;

      line='}' ;

      output ;

      end ;

**run** ;