Paper 84

SAS/GRAPH® Software via RSUBMIT for Web Browser

Hsiwei Yu (Michael), Digital Intelligence Systems, Centreville, VA

ABSTRACT

This paper discusses techniques for generating SAS/Graph® on remote host, then sending it back to the web server machine for display on the Internet.

INTRODUCTION

Problem: SAS/IntrNet® is licensed on local host, but huge amount of corporate data resides on remote host, say MVS®. How to produce graph based on remote data for Internet display dynamically and without waiting forever?

With SAS/Connect® licensed, we can, from a local interactive SAS session, remote submit programs to generate graph for display on local host. However, if the local SAS session is being run in batch mode or under SAS/IntrNet's control, then changes are needed to properly retrieve graphs from the remote host. Secondly since textual or graphical reports are usually smaller in size than the actual data, it's better to produce report on the host where the data resides, then to transfer only the report across network.

WHY NOT USE REMOTE LIBRARY SERVICE?

For example,

```
Libname libref server=
remote_host.SAS_Share_server;

Goptions device= gif gsfname= _webout;
Proc gchart data= libref.SAS_data;
Run;
```

Remote library service, by definition, brings data from remote host down to local host for processing. In most enterprise, network speed is the bottleneck. With SAS/Connect, we can take advantage of the remote host's processing power while transfer only a fraction of data across network. Also when many simultaneous requests coming from the Internet, either IntrNet or Share® servers may become saturated, causing difficulty to isolate the root cause of poor response time.

WHAT IS DIFFERENT FROM AN INTERACTIVE REMOTE SUBMIT?

When running interactively, remote submit generates a graph and displays it in a local graph window. But when the local SAS session is being run in batch mode, or is under SAS/IntrNet's control, graph window is not available. Under these circumstances, setting the local SAS session's goptions 'nodisplay' properly suppresses the graph window, but unfortunately prevents the graph from being sent back to the local SAS session. So we must add a 'proc download' step to retrieve the graph from the remote host. Finally if running under SAS/IntrNet's control, a 'proc greplay' is needed to display the graph to the Internet.

```
rsubmit;
goptions device= gif;

proc gchart data= .. ; run; quit;

proc download incat= work.gseg outcat= work.gseg
     status= n;
run;
endrsubmit;

/* Since local host has SAS/Graph, we can use
Proc Greplay */
goptions display device= gif gsfname= _webout;
proc greplay igout= work.gseg nofs;
replay gchart;
run;
quit;
```

WHAT IF LOCAL HOST DOES NOT HAVE SAS/GRAPH?

Should the local host not have SAS/Graph, the remote host can produce a file in GIF format, then transfer this file in binary mode to local machine for immediately display.

```
filename local
"%sysfunc(pathname(work))\abc.gif";

rsubmit;
filename rmt_gif '&abc' unit=sysda
space=(trk,(5,5));
goptions device= gif gsfname= rmt_gif
gsfmode= replace;

proc gchart data= ..; run; quit;

proc download infile= rmt_gif
              outfile= local
              status= no binary;
run;
endrsubmit;

data _null_;
  file _webout;
infile local;
 input ;
   put _infile_;
run;
```

OTHER CONSIDERATIONS

The variable 'alias' in dictionary.catalogs or equivalently, sashelp.vcatalg view, can help us identify the mostly recently generated SAS/Graph entry. This parameter is needed by proc greplay to identify the graph being reshown.

SAS/Connect also supports the 'spawner' program on some platforms. With spawner program running as server, a client can request work from the spawner without signon and signoff. Consult the 'Communication Access Methods for SAS/Connect and SAS/Share Software' manual for details.

Be sure to hide your userid and password when using SAS/Connect script file for remote signon and signoff in batch mode. Don't store the userid or password as macro variables, since they can be displayed simply by

```
%put _ALL_;
```

A TCP/IP SOCKET SERVICE ALTERNATIVE

To save the overhead of starting up a new SAS session on the remote host on demand, a SAS session running on the remote host can be setup to wait for service requests coming from a TCP/IP socket port. For details see the paper titled "Using Sockets in SAS Software for Internet Publishing" by this author.

CONCLUSION

Assuming your organization has SAS running on multiple platforms, but SAS/IntrNet is only licensed on one of them, say PC. However you're not limited or obligated to store all

the data and produce all the web-based reports from this single environment. By taking advantage of the SAS System's cross platform capability, other data sources can just as easily be published to the Internet.

ACKNOWLEDGMENTS

This author likes to thank Chapman Gleason of US Environmental Protection Agency and Myles Powers of Logicon for formulating this requirement, and for their support and encouragement for its solution.

CONTACT INFORMATION

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

Hsiwei Yu (Michael)
Digital Intelligence Systems
Centreville, Virginia
http://www.disys.com
Work Phone: 202-260-5312
Fax: 202-260-4968
Email: vhyu@netkonnect.com

SAMPLE CODE WITH PROC GREPLAY

```
goptions nodisplay;

options comamid= tcp remote= ceisserv;
filename rlink
'!sasroot\connect\saslink\tcpunix2.scr';
signon;

rsubmit;
libname a '~myu/sasuser';

goptions device= gif;

proc gchart data= a.crime;
vbar staten / sumvar= auto;
run;
quit;

proc download incat= work.gseg outcat= work.gseg
     status= n;
run;

endrsubmit;

signoff;

filename rlink;

goptions display device= gif gsfname= _webout;
proc greplay igout= work.gseg nofs;
replay gchart;
run;
quit;
```

SAMPLE CODE USING BINARY TRANSFER OF GIF FILE

```
filename local
"%sysfunc(pathname(work))\abc.gif";

options comamid= tcp remote= epaibm;

filename rlink
'!sasroot\connect\saslink\tcptso-for-
ibm.scr';
signon;

rsubmit;

/* Creating a file of GIF format on the
remote host for later transfer */
filename remote '&temp' unit=sysda
space=(trk,(5,5));
goptions device= gif gsfname= remote
gsfmode= replace;

proc gchart data= sasuser.crime;
vbar staten / sumvar= auto;
run;
quit;

proc download  infile= remote
              outfile= local
               status= no binary;
run;

endrsubmit;

signoff;

/* Since SAS/Graph not available on local
host, simply sends the GIF file to the web
browser for display */
data _null_;
  file _webout;
infile local;
 input ;
   put _infile_;
run;

filename local;
```