Paper 286-25

Master Your Domain: Automated Software Distribution In A Client/Server Environment

Jeff Lessenberry, Jeff Lessenberry Consulting Group

Abstract

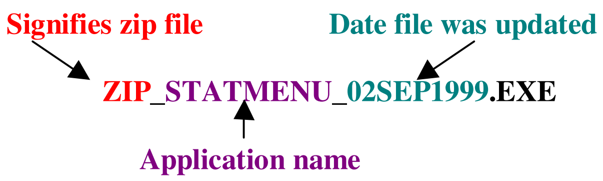
So many application changes but so little time. This paper will address the problem of distributing software to users across the hall or across the country. As companies become more departmentalized, the red tape involved in adding or updating an application grows exponentially. Just as too many cooks can spoil the soup, dealing with multiple server administrators and installation personnel can cause major synchronization problems. By automating the software distribution process, the user can have the latest and greatest software with a transparent update process. The techniques discussed in this paper utilize current day technology rather than vaporware.

Introduction

Some of us old timers might remember when the worlds' biggest fear was nuclear war not Y2K and the only computers were mainframes. Distribution of software was much simpler when everyone was using a dumb terminal and all of the applications resided on one machine. The creation of the personal computer brought control back to the user and made Bill Gates millions in software license fees. Users soon realized they had lost access to their data so client/server systems were born. The development of the client/server system also created a centralized location for software distribution. This paper will show one way to exploit the client/server platform to distribute software.

Preparing the Files

The first step is to pick a naming convention that will allow the application files to be chosen for update. The following naming convention to worked best for my needs:



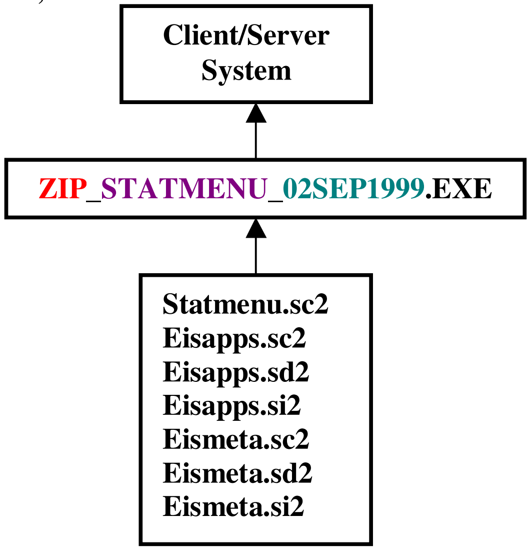
By choosing the previous naming convention, the application can determine which files are current and which files need to be updated.

The upload file is in a self-extracting zip file for several reasons:

1. You can zip all files needed by an application like application catalogs, SAS/EIS files, and data templates.
2. If the application frame is filled with a tab layout object and also contains a tab layout object on one of the main tabs it can not be uploaded using the PROC UPLOAD. If this process is attempted to server connection will crash.
3. The most important reason is SPEED. The size of a zip file is much smaller than the individual files. Your network administrator will thank you for decreasing the network traffic.

The self-extracting zip file must be uploaded to the client/server box using PROC UPLOAD. The format of the PROC UPLOAD process is the following:

```
Filename lziprefs c:\apps\enus\rsas612\proj\unixzips';
Rsubmit;
Filename rzipref '/grq/busdata/rgrq/mna1/enus/user/apps';
   proc upload infile  = lziprefs(zip_vector_09aug1999.exe)
                outfile =  rzipref(zip_vector_09aug1999.exe)
          binary ;
    run;quit;
Endrsubmit;
```



Running the Process

If the application runs at multiple sites, only two items must be distributed to each site for the startup process; a startup icon and a SAS initialization program. The initialization program defines the applications libraries and sets SAS environmental options. The startup command line is the following:

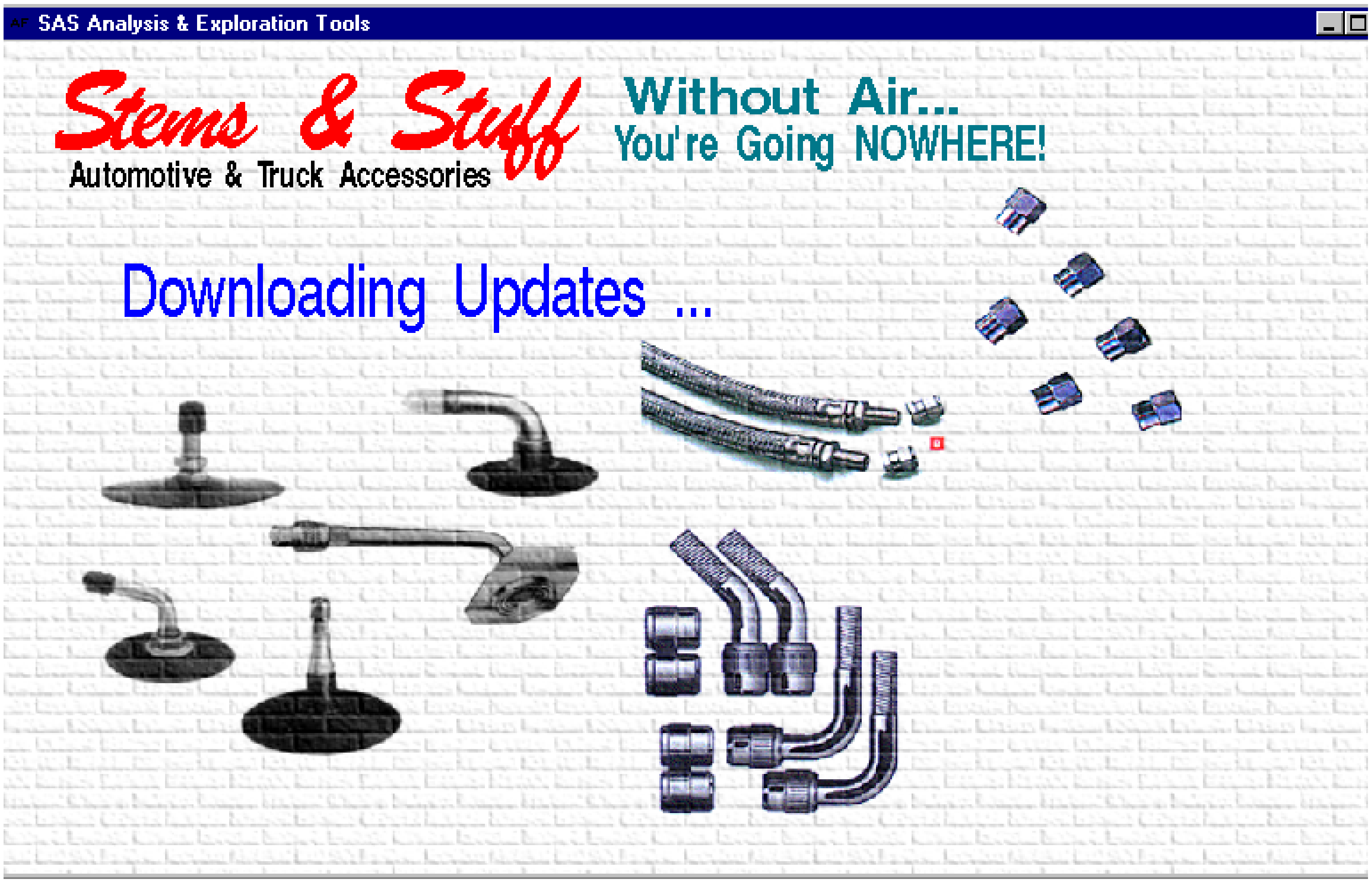
```
C:\SAS\sas.exe -autoexec c:\SAS\strtmenu.sas
-initcmd "af c=menuproj.strtmenu.start.frame"
-awscontrol notitle -awsdef 0 0 100 100
```

The following is an example of the initialization files to start the application:

```
options noxwait;
libname menuproj 'c:\statmath\solution\apps';
options font='Sasfont' 8;
```

A download screen is displayed to keep the user occupied while the application checks the version numbers and updates old files.

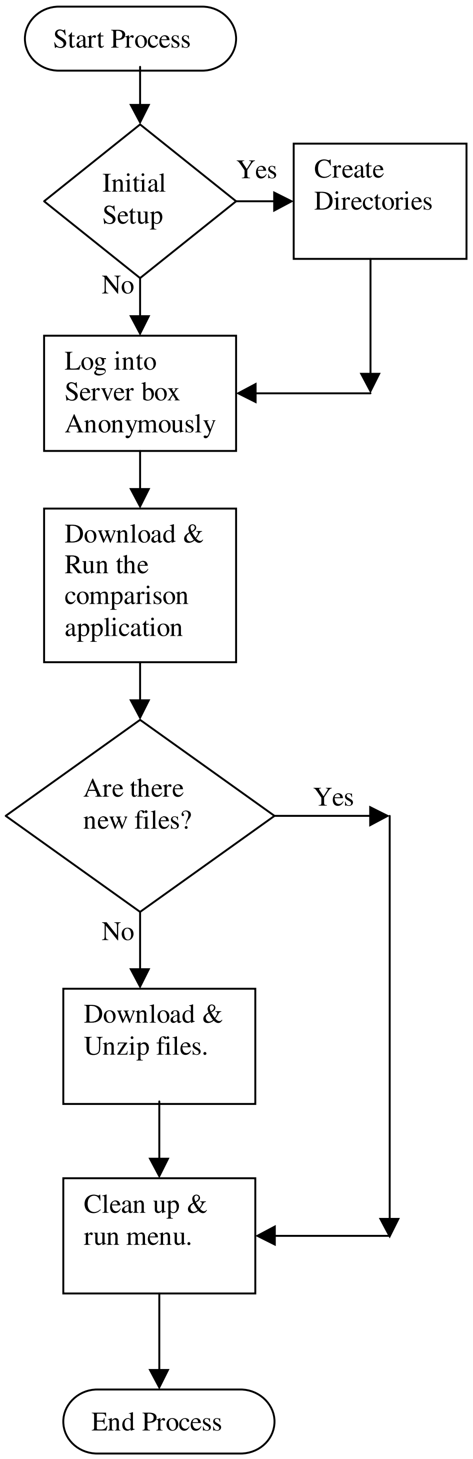
STRTMENU.FRAME

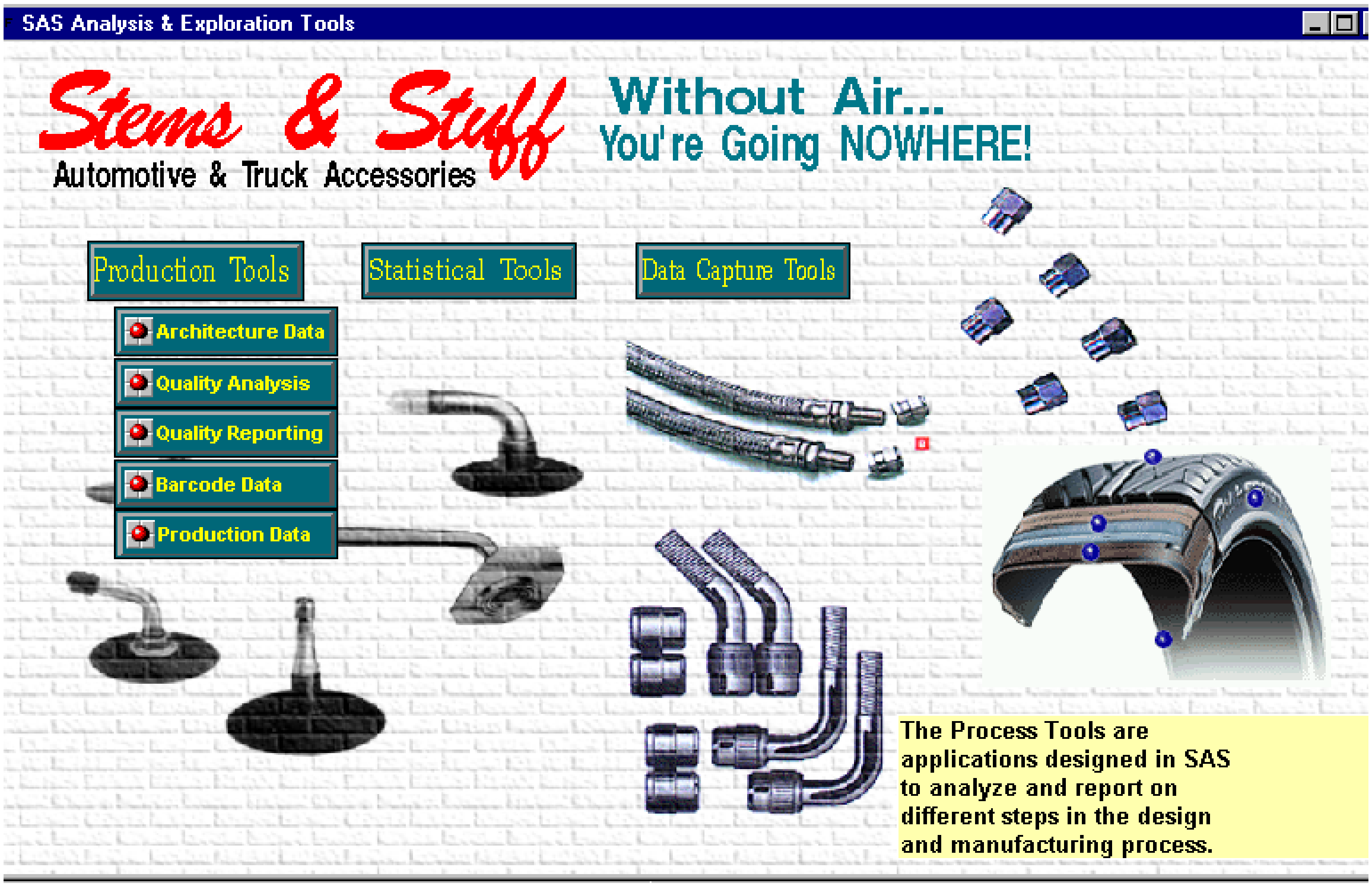


STRTMENU.SCL

```
INIT:
   control asis;

/*      Define Server Name         */

   server='oncpltp1';

/*   Check for application path    */

   ex = fileexist('c:\statmath\solution');
   if ex = 0 then do;
      rc= system('md c:\statmath');
      rc= system('md c:\statmath\solution');
      rc= system('md c:\statmath\zipfiles');
      rc= system('md c:\statmath\sasfiles');
   end;
   CALL send(_self_,'_refresh_');

/* Create login script at runtime (Security) */

   link SCRPTSET;

/*  Login to server using anonymous login  */

   Submit Continue;
      filename rlinklog 'THELOGFILE.scr';
      proc printto log=rlinklog new;
      run;
      filename rlink 'THESCRIPT.scr';

      options comamid = TCP
      remote = &server;

      signon rlink;
      libname remapps slibref=appslib
              server=&server;;
   EndSubmit;

   rapps = pathname('remapps');
   Submit Continue;
      libname remapps ;

/* Download comparison application to PC */

      rsubmit;
         proc download incat=appslib.download
                    outcat=work.download
            status=N;
         run;

/*        Run comparison remotely          */

         proc display
              c=appslib.download.remchk.scl;quit;
      endrsubmit;

      libname rwork slibref=work server=&server;

/*        Run comparison locally          */

      proc display c=work.download.localchk.scl;quit;

/*  Sort and Merge result files - keep new only  */

      proc sort data=loczips;
           by appname;
      run;
      proc sort data=rwork.remzips;
           by appname;
      run;
      data allzips;
           merge work.loczips rwork.remzips;
           by appname;
           if remdate = locdate then delete;
      run;
      filename lzipref 'c:\statmath\zipfiles';
      rsubmit;
         filename rzipref '&rapps';
      endrsubmit;
   EndSubmit;

/*        Load file names into SCL lists   */

   zipfile = open('allzips');
   if attrn(zipfile,'NLOBS') ne 0 then do;
      call notify('chkver','_set_text_',
                  'Downloading New Versions ....');
      CALL send(_self_,'_refresh_');
      lziplist=makelist();
      rziplist=makelist();
      lev=0;
      rc=lvarlevel(zipfile,'lzipname',lev,lziplist);
      rc=lvarlevel(zipfile,'rzipname',lev,rziplist);

/*        Delete old zip files              */

      do it = 1 to listlen(lziplist);
         delfile = getitemc(lziplist,it);
         if delfile ne '' then do;
            rc=filename
         ('delref',"c:\statmath\zipfiles\"!!delfile);
            rc=fdelete('delref');
            rc=filename('delref','');
         end;
      end;

/*      Setup zip file list             */

      zipsrun = makelist();
      do it = 1 to listlen(rziplist);
         copyfile = getitemc(rziplist,it);
         cpyfile =
                "c:\statmath\zipfiles\"
                !!trim(left(copyfile))!!" -d -o c:\";
         rc=insertc(zipsrun,cpyfile,-1);
```

```
 /*       Download zip files          */

      Submit Continue;
           rsubmit;
             proc download infile=rzipref(&copyfile)
                         outfile=lzipref(&copyfile)
                binary status=N;
             run;quit;
           endrsubmit;
      EndSubmit;
   end;

/*  Create and run batch file for zip extraction   */

  rc=savelist('FILE','RUNZIPSBATFILE.BAT',zipsrun);
      Submit Continue;
         x 'runzipsbatfile.bat';
      EndSubmit;
      rc=filename('delref',"runzipsbatfile.bat");
      rc=fdelete('delref');
      rc=filename('delref','');
   end;

/*      Define library and clean up (Security)     */

   Submit Continue;
      libname proj 'c:\statmath\solution\apps';
      signoff;
      proc printto;
      run;
      dm 'clear recall';
   EndSubmit;
   drc=fdelete('rlink');
   drc=fdelete('rlinklog');
   rc=filename('rlink','');
   rc=filename('rlinklog','');

/*               Run Menu Screen               */

   call display('proj.statmenu.menu.frame');
   _status_='H';
return;
```





The Process Tools are applications designed in SAS to analyze and report on different steps in the design and manufacturing process.

Benefits

Automated software distribution makes updates painless for the developer and transparent for the users. The above menu screen is an example of how applications can be added or removed from the selections by updating one centrally located frame. The user will automatically receive the updated menu the next time the application is activated. Synchronization is not an issue since all users will get the updated files. This is an important benefit, if the application runs at several sites in multiple countries. This returns application control to the developer and decreases the amount of bureaucracy that is involved in updating an application.

User tracking is another benefit of using this process. Tracking which applications the users are running has many benefits. Future application update decisions can be made using this feature. If an application is only used three times in six months, should money be allocated toward enhancing it? Knowing the volume of users can be useful in determining the effect on network throughput. If the usage numbers increase, this can show cause for allocating a better network line or even the purchase of a new server box.

Conclusion

The faster changes can be fixed or new applications installed, the happier the client will be. This process uses current SAS processes to accomplish these tasks. I am the first person to ooh and ah over the new shinny toy but I will settle for an older one that works well now. Only the time can reveal what products will be introduced in the future but I have found nothing beats a little thing called - creativity.

Acknowledgements

SAS, SAS/EIS, SAS/AF, SAS/CONNECT and all other SAS references are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Jeff Lessenberry
Jeff Lessenberry Consulting Group
Telephone: (864) 243-9761
Email: JLCG@Mindspring.com