Mainframe-based PC Software Maintenance

Cathy Kilburn, SAS Institute Inc., Cary, NC

ABSTRACT

The Education Division at SAS Institute uses personal computers for student workshops in its training centers around the country. We must assure that all the proper files are on these machines each time a student takes a class. Therefore, our goal was to develop a system to easily maintain dozens of PCs from a central location, namely the mainframe computer in Cary, North Carolina. This application relies on base SAS® software on both the mainframe and the PC, especially the UPLOAD and DOWNLOAD procedures.

We use the mainframe to store copies of all the PC files (in binary format) along with an associated master list that identifies all the files that should be on the machines in any given training center at any given time. Then, when maintenance is run on a machine, the system uses the SAS System to upload to the mainframe a complete directory listing of the files currently on the PC. This list is compared to the master list and any missing or modified PC files (based on a comparison of the number of bytes, and the date and time of the last update) are replaced by downloading the mainframe copy of the file. In addition, any unauthorized files found on the PC are erased.

INTRODUCTION

The Problem

This paper presents a technique that can simplify the job of managing a large number of machines, especially when they are at remote sites. Through this technique, PC software can be maintained via a mainframe connection. The Education Division at SAS Institute uses personal computers in its training centers around the country. These machines can be used either as stand-alone PCs or as mainframe terminals. They are used primarily by students to run the workshops that accompany SAS training courses. They are also used occasionally by SAS Marketing Division personnel to demonstrate SAS software products to customers. Since these machines are used by many different people for different purposes, there is a large potential for intentional and unintentional alteration of the files needed for student workshops.

We keep all the necessary SAS software, workshop data, terminal emulation software, and so on, on the hard drive of each machine. Thus, we have the quality control challenge of assuring that each machine is in proper working order (that is, that all necessary files are available) each time a student comes in to take a class. We also must occasionally update the SAS software or the data used for class workshops, so we need a way to distribute new files to the PCs as needed. The job of maintaining the software on these PCs is complicated by the number of machines involved (over 100) and the fact that they are located all over the country, while the people responsible for this maintenance are located at SAS Institute headquarters in Cary, North Carolina. In addition, the hardware and software differs somewhat from training center to training center, making it necessary to treat each one a little differently.

Our Goals

Our objective was to develop a system for PC software maintenance that could be easily administered from a central location (Cary) and would assure standardization of all machines in a training center. The system should be able to determine quickly and automatically which files are missing or altered on a given PC, based on comparison with a predefined standard, and then be able to correct the machine (add, remove, or replace files) as needed. This approach relieves staff members in the training centers of the tedious and error-prone tasks of browsing dozens of subdirectories to find out what is missing or altered and then copying needed files from diskette.

The system should also allow for files or whole directories to be exempt from maintenance so users could keep personal files on the PC (as long as they are properly listed in our master files).

The system should be easy for nontechnical staff members in the training centers to use; each training center is staffed by a full-time course coordinator who will eventually be responsible for running the system. The system must give us the flexibility to define different standards for each training center to allow for differences in specific terminal emulation software. And the system should allow us to easily modify the standard and then quickly update all the machines in a given training center.

IMPLEMENTATION

Overview of the PC Maintenance System

The PC Maintenance System (PCMAINT) uses SAS micro-to-host link software to allow communication between the PCs in our training centers and the IBM mainframe in Cary (SAS Release 6.03 on the PC and Release 5.18 on the mainframe are required). We store copies of all the PC files on the mainframe; there is also a master list on the mainframe that identifies all the files that should be on the PCs in any given training center at any given time. Then, when PCMAINT is run on a machine, the system uses PROC UPLOAD to send a complete list of the files currently on the PC to the mainframe. Upon processing this list of PC files, the system can handle all of these specific needs:

- erase unauthorized files
- create new directories
- remove unauthorized directories
- download missing files
- download altered files
- allow files or whole directories to be exempt from maintenance.

Master Lists and Mainframe Copies

The first step in setting up the PCMAINT system was to define the master list of files for each training center and transfer a copy of each PC file to the mainframe. The master lists are in SAS data sets on the mainframe, with one data set per training center. Each data set contains one observation per PC file, and each observation includes the complete PC file specification (path and file name), the size of the file (in bytes), and the name of the mainframe file containing a copy of the PC file (Table 1). It also contains the variable INSTRUCT, which is the instruction for what should be done to this file (or directory); the only two values are COMPARE and EXEMPT.

Table 1 Excerpt of a SAS Data Set Containing a Master List

DIR	FNAME	FEXT	BYTES	MVSFILE	INSTRUCT
C:\DOS					COMPARE
C:\DOS	CHKDSK	COM	9850	'PCM.DOS.DIR(CHKDSK)'	COMPARE
C:\DOS	DR	COM	3370	'PCM.DOS.DIR(DR)'	COMPARE
C:\DOS	EDLIN	COM	7526	'PCM.DOS.DIR(EDLIN)'	COMPARE
C:\DOS	FORMAT	COM	11616	'PCM.DOS.DIR(FORMAT)'	COMPARE

The PC files are stored in binary image form on the mainframe; that is, an exact copy of the PC file is made, with no ASCII to EBCDIC conversion. There is no reason to convert since the files are not actually used in any way on the mainframe. In fact, since the sole purpose of these files is to be eventually transferred back to a PC, it would be inefficient to convert each time.

The copies are kept as members of partitioned data sets (PDS), which is helpful for grouping files that come from the same directories on the PC. The file characteristics of the PDSs were chosen to optimize data transfer between the PC and mainframe when using PROC DOWNLOAD. A record length (LRECL) of 256 was chosen, as this is the largest length supported by the procedure. A large block size (BLKSIZE=23000) was chosen to reduce the number of I/O requests that the SAS System must make to MVS. Variable-length records (RECFM=VB) were chosen to prevent padding of the last record with blanks, which would occur with fixed-length records.

Mainframe Accounts

The PCMAINT system requires a TSO logon ID for making a connection to the mainframe. To allow many machines to run PCMAINT at once, we decided to devote one TSO logon ID to each PC. This reduces the amount of time it takes to maintain all the PCs in a training center.

Step-by-Step Summary of PCMAINT

With the master lists, copies of PC files, and TSO accounts in place on the mainframe, we were ready to install and run the PCMAINT software on the PCs. To install PCMAINT on the PC, we put the four files needed for the system in the D:\PCM subdirectory:

PCMAINT.BAT	DOS batch file that starts the system
MAINT.SAS	SAS program called by PCMAINT.BAT
MAINT2.SAS	SAS program called by MAINT.SAS, which further calls the mainframe SAS program
MAINT.SCR	Script file that controls the micro-to-host link.

To run PCMAINT, the user simply changes into the D:\PCM directory and types PCMAINT; no further user intervention is required. This section traces the sequence of steps that occur when the system is invoked.

1. PCMAINT.BAT
   This bat file produces a directory listing of all the files currently on the machine; this listing is directed to a file for future reference. Then the terminal emulation software and the SAS System are started. We call the SAS System with an AUTOEXEC file, MAINT.SAS, which executes next.

2. MAINT.SAS (Appendix 1)
   This SAS program includes the statements necessary to set up the remote link and to submit statements to the mainframe automatically. The included program, MAINT2.SAS, actually does the remote signon and remote submit. Automated remote processing is documented in SAS Technical Report P-187, *Changes and Enhancements to the SAS Micro-to-Host Link, Releases 5.18 and 6.03.*Our script file for the remote link is a slightly modified version of the LOGTSO.SCR script which is distributed with PC SAS software. In the 3270 SIGNON section of the script, we use the following statement to logon automatically to a specific TSO account which has been assigned to this PC (xxx indicates a particular account, for example, PCM001, PCM015):

   ```
   type 'tso PCMxxx';
   ```

3. MAINT2.SAS (Appendix 1, Step 1)
   This program is included and submitted by MAINT.SAS. Step 1 of the program is a group of SAS statements that is submitted to the mainframe by enclosing it in an RSUBMIT/ENDRSUBMIT block. There are just three statements here, the minimum required to get the mainframe SAS program running. One of our design goals was to put as much mainframe code as possible on the mainframe itself, rather than on the PC; this allows us to easily change the code if necessary, without having to distribute new system diskettes to all the training centers. The first statement allocates the file containing the mainframe code that we want to run at this point. We set the SITE macro variable in a %LET statement so the next program will know which master list to use for this machine. Finally, the MAINT program is run with a %INCLUDE statement.

4. 'PCM000.MAINT.SAS(MAINT)' (Appendix 2)

   □ Step 1 - This program starts by calling a CLIST to allocate the mainframe files needed by the application. It is very helpful to put these allocations in a CLIST because, if an allocation is issued directly from the SAS System and it fails, a message will be written to the terminal, which must be cleared before execution can continue. Thus, the user would need to monitor the PC and switch to the mainframe session to clear the message if this situation occurs. However, by putting the allocations in a CLIST with a CONTROL NOMSG statement, you can avoid this problem. Of course, normally you would not want to ignore such messages, but we have several allocations that may fail if the file doesn't already exist (we anticipate this situation by checking a return code and creating the file if necessary).

   □ Step 2 - This part of the program uploads the PC file list and creates a SAS data set with the following variables:

DIR	$ 50	directory name (path)
FNAME	$ 8	filename
FEXT	$ 3	filename extension
BYTES	$ 8	number of bytes in file
DATE	$ 10	date of last change to file
TIME	$ 11	time of last change to file

   Notice that DIR, FNAME, and FEXT are also found in the master data sets; these fields will be used to match observations. The SWEEP SAS data set contains one observation per PC file; there is also a separate observation for each subdirectory, with missing values for FNAME and FEXT. These special *directory-only* observations will be used to take various actions on a directory level, such as creating or removing directories.

Finally, the SWEEP data set is sorted by DIR, FNAME, and FEXT.

□ Step 3 - In the next several DATA steps, we preprocess the SWEEP data to accomplish several things:

- If an entire directory is marked EXEMPT in the master list, remove all files in that directory from the SWEEP data set (this feature makes it easy to exempt many files at once, without having to identify them all in the master list).
- Create a SAS data set containing a list of directories that need to be created.
- Create a SAS data set containing a list of directories that need to be removed.

We first divide the master list for the given training center into three data sets: one containing a directory-only observation for exempt directories, one containing a directory-only observation for directories to be compared, and one containing the rest of the original observations, that is, all the observations for individual files.

In the next DATA step, we use the SET statement to bring in one exempt directory at a time and then use the SET statement with the POINT option to search the entire SWEEP data set for files in that directory; this allows us to remove those files selectively from SWEEP. The final DATA step merges the COMPARE directories (DIRSCOMP) with the SWEEP data set, using the special IN= variable to determine whether the current observation is coming from both data sets (a matching DIR was found) or just one of them. If the directory is only found in DIRSCOMP and not in SWEEP, that directory should be on the PC but is not. In that case, an observation is output to MKDIR. On the other hand, if the directory is found only in SWEEP, it is unauthorized, and an observation is output to RMDIR.

The MKDIR and RMDIR data sets will be used later to construct a PC bat file to do the actual MKDIR (make directory) and RMDIR (remove directory) commands. Notice also that, at the end of this step, SWEEP contains only the observations for files; the directory-only observations have been removed.

□ Step 4 - In this DATA step, we merge the SWEEP and FILES data sets, again using IN= variables. As we do the merge, one of the following situations will occur for the current observation:

- The file is EXEMPT (there may still be exempt files that are not within exempt directories); no action is taken.
- The file is found on both the PC and master list, but the PC file has been altered; in this case, a DOWNLOAD statement is written to the flat file DOWNLOAD. This file will accumulate all the DOWNLOAD statements for later execution.

  The PC file is tested on three criteria to determine if it has been altered. First, if the size of the file differs from its description in the master list, then obviously the contents have been changed. However, the file could have been altered but still have the same number of bytes. So we also check the date and time of the file; if it differs from a standard that we have previously set on all the files (1-01-80, 12:00a), we assume that file has been changed. Now, when a file is downloaded, DOS will set its datetime stamp to the current value on the PC. So we must reset it to our standard datetime to avoid unnecessary downloading in the future. We do this via a C program that uses a special function to reset a file's date and time. This explains the additional DATA step and PROC DOWNLOAD step following the PROC DOWNLOAD of the original file. These two steps allow us to accumulate a list of all the files that have been downloaded; this list will later be used as input to the C program.
- The file is on the master list but not on the PC; as in the above case, two DOWNLOAD statements are generated: one that will download the original file and one that will download the list of files to be reset (datetime).
- The file is on the PC but not on the master list; we write a DOS ERASE command to the flat file that will be downloaded and executed as a bat file on the PC.

□ Step 5 - Now we must add any MKDIR and RMDIR commands to the DOS bat file. The SAS data sets MKDIR and RMDIR are first sorted by directory; then the commands are written to the DOS file in two DATA steps. Finally, the file is downloaded for execution on the PC.

5. MAINT2.SAS (Appendix 1, Steps 2 & 3)
Now control returns to the PC, where we are still running the SAS program MAINT2.SAS (the remote link is still active, however). Now we actually do all the work that was prescribed in the preceding steps. The MAINT.BAT file is executed to get rid of unwanted files and directories and to create any new directories needed. Then the file containing all the DOWNLOAD statements is run with an RSUBMIT command. When the downloads are finished, the remote link is ended.

6. PCMAINT.BAT
When MAINT2.SAS is finished, control is returned all the way back to the original bat file. To finish up, we run the C program, which resets the date and time on all downloaded files, and erase three temporary files that we no longer need.

APPENDIX 1. PC SAS CODE

```
MAINT.SAS

       options remote=irma;
       filename rlink "d:\pcm\maint.scr";
       dm 'inc "maint2.sas"';
       dm 'submit';

MAINT2.SAS

DM 'SIGNON';
  /********************  STEP 1  ***********************************/
  /* Submit mainframe code to upload and process sweep data        */
  /*    ("clean up" and compare to master file list)               */
  /****************************************************************/
  DM 'RSUBMIT';
    X ALLOC F(MF) DA('PCM000.MAINT.SAS') SHR;
    %LET SITE=CA;
    %INC MF(MAINT);
  ENDRSUBMIT;
  /********************  STEP 2  ***********************************/
  /* Run DOS batch file to do any RMDIRs, MKDIRs, and ERASEs       */
  /****************************************************************/
  DM "X 'D:\PCM\MAINT.BAT'";
  /********************  STEP 3  ***********************************/
  /* Submit PROC DOWNLOADs                                         */
  /****************************************************************/
  DM 'RSUBMIT';
    TSO ALLOC F(DOWNLAOD) DA(DOWNLANO.PGM) SHR REU;
    %INC DOWNLOAD;
    TSO FREE F(DOWNLOAD);
  ENDRSUBMIT;
DM 'SIGNOFF';
```

APPENDIX 2. MAINFRAME SAS CODE (MAINT)

```
/************************  STEP 1  *******************************/
/* Execute CLIST to allocate necessary files                     */
/****************************************************************/
TSO PCMSTART;
/************************  STEP 2  *******************************/
/* UPLOAD and "clean up" list of PC files                        */
/****************************************************************/
PROC UPLOAD INFILE='D:\PCM\SWEEP.DAT' OUTFILE=RAWSWEEP;
DATA SWEEP(KEEP=DIR FNAME FEXT BYTES DATE TIME);
  INFILE RAWSWEEP;
  LENGTH DIR $ 50;
  INPUT @1 LINE $80. @;
  LINE=UPCASE(LINE);
  IF LINE=' ' THEN DELETE;
  ELSE IF INDEX(LINE,'<DIR>') > 0 THEN DELETE;
  ELSE IF INDEX(LINE,'DIRECTORY OF') > 0 THEN DELETE;
  ELSE IF INDEX(LINE,'VOLUME IN') > 0 THEN DELETE;
  ELSE IF INDEX(LINE,'BYTES FREE') > 0 THEN DELETE;
  ELSE IF INDEX(LINE,'>>> SWEEP >>>') > 0 THEN DO;
    DIR=UPCASE(SUBSTR(LINE,15));
    RETAIN DIR;
  END;
  ELSE DO;
    INPUT FNAME $ 1-8 FEXT $ 10-12 BYTES 14-23
          DATE $ 24-33 TIME $ 34-44;
    FNAME=UPCASE(FNAME);
    FEXT=UPCASE(FEXT);
    TINE=UPCASE(TIME);
  END;
RUN;
PROC SORT DATA=SWEEP;
   BY DIR FNAME FEXT;
RUN;
/************************  STEP 3  *******************************/
/*  Break master data set into three data sets: one for files,   */
/*  one for exempt directories, and one for directories to be    */
/*  compared. Remove files in exempt directories from SWEEP.     */
/*  Create list of files to be either removed or created.        */
/****************************************************************/
DATA FILES(KEEP=DIR FNAME FEXT BYTES INSTRUCT MVSFILE)
     DIRSEXEM(KEEP=DIR)
     DIRSCOMP(KEEP=DIR);
  SET MASTER.&SITE;
  IF FNAME=' ' AND FEXT=' ' THEN DO;
    IF INSTRUCT='COMPARE' THEN OUTPUT DIRSCOMP;
    ELSE IF INSTRUCT='EXEMPT' THEN OUTPUT DIRSEXEM;
  END;
  ELSE IF FNAME¬=' ' THEN OUTPUT FILES;
RUN;
DATA SWEEP (KEEP=DIR FNAME FEXT BYTES DATE TIME);
  SET DIRSEXEM(RENAME=DIR=MASTDIR);
  DO I=1 TO NOBS;
    SET SWEEP POINT=I NOBS=NOBS;
    IF INDEX(TRIM(DIR),TRIM(MASTDIR)) = 1 THEN;
    ELSE OUTPUT;
  END;
RUN;
DATA SWEEP (KEEP=DIR FNAME FEXT BYTES DATE TIME)
     MKDIR (KEEP=DIR)
     RMDIR (KEEP=DIR);
  MERGE DIRSCOMP(IN=ONMASTER)
        SWEEP(IN=ONSWEEP);
  BY DIR;
  IF ONSWEEP AND FNAME¬=' ' THEN OUTPUT SWEEP;
  IF ¬ONSWEEP THEN DUTPUT MKDIR;
  IF ¬ONMASTER AND FNAME=' ' AND FEXT=' ' THEN OUTPUT RMDIR;
RUN;
/************************  STEP 4  *******************************/
/* Process the remaining SWEEP observations and create the list  */
/* of DOS ERASE commands, the list of PROC DOWNLOAD statements,  */
/* and a log of all actions taken.                               */
/****************************************************************/
DATA _NULL_;
  MERGE FILES(IN=ONMASTER RENAME=(BYTES=BYTES2))
        SWEEP(IN=ONSWEEP) END=EOF;
  BY DIR FNAME FEXT;
  IF _N_=1 THEN DO;
    FILE MAINTLOG;
    PUT 15*'-' "&SYSDATE  &SYSTIME " 15*'-';
  END;
  IF INSTRUCT='EXEMPT' THEN RETURN;
  IF DIR='C:\' OR DIR='O:\' THEN
    PCFILE=TRIM(DIR) || TRIM(FNAME) || '.' || TRIM(FEXT);
  ELSE
    PCFILE=TRIM(DIR) || '\' || TRIM(FNAME) || '.' || TRIM(FEXT);
  QPCFILE= "'" || TRIN(PCFILE) || "'";
  IF INSTRUCT='COMPARE' AND ONMASTER AND ONSNEEP THEN DO;
    IF BYTES¬=BYTES2 OR DATE¬='1-01-80' OR TIME¬='12:00A' THEN DO;
      FILE DOWNLOAD;
      PUT 'PROC DOWNLOAD INFILE=' MVSFILE;
      PUT "                  OUTFILE='" PCFILE "' BINARY; RUN;";
      PUT 'TSO ALLOC F(PCFILES) DA(DOWNLOAD.PCFILES) MOD REU;';
      PUT 'DATA _NULL_; FILE PCFILES; PUT ' QPCFILE '; RUN;' ;
      PUT 'PROC DOWNLOAD INFILE=PCFILES';
      PUT "                  OUTFILE='D:\PCM\CHGDATE.DAT'; RUN;";
      PUT 'TSO FREE F(PCFILES);';
      PUT;
      FILE MAINTLOG;
      PUT MVSFILE ' DOWNLOADED TO  ' PCFILE;
    END;
  END;
  ELSE IF INSTRUCT='COMPARE' AND ONMASTER AND ¬ONSWEEP THEN DO;
    FILE DOWNLOAD;
    PUT 'PROC DOWNLOAD INFILE=' MVSFILE;
    PUT "                  OUTFILE='" PCFILE "' BINARY; RUN;";
    PUT 'TSO ALLOC F(PCFILES) DA(DOWNLOAD.PCFILES) MOD REU;';
    PUT 'DATA _NULL_; FILE PCFILES; PUT ' QPCFILE '; RUN;';
    PUT 'PROC DOWNLOAD INFILE=PCFILES';
    PUT "                  OUTFILE='D:\PCM\CHGDATE.DAT'; RUN;";
    PUT 'TSO FREE F(PCFILES);';
    PUT ;
    FILE MAINTLDG;
    PUT NVSFILE ' DOWNLOADED TO  ' PCFILE;
  END;
  ELSE IF ¬ONMASTER AND ONSWEEP THEN DO;
    PUT 'ERASE ' PCFILE;
    FILE MAINTLOG;
    PUT  'ERASE ' PCFILE;
  END;
RUN;
TSO FREE F(DOS MAINTLDG);
/************************  STEP 5  *******************************/
/*  Sort the MKDIR and RMDIR data sets and append any MKDIRs and */
/*  RMDIRs to the DOS.BAT file. Download this file to be run on  */
/*  the PC. The PROC DOWNLOADs will be run (via RSUBMIT) after   */
/*  the ERASEs, MKDIRs, and RMDIRs are done.                     */
/****************************************************************/
PROC SORT DATA=MKDIR NODUPS;
  BY DIR;
```

```
RUN;
PROC SORT DATA=RMDIR NODUPS;
  BY DESCENDING DIR;
RUN;
TSO ALLOC F(DOS) DA(DOS.BAT) MOD REU;
TSO ALLOC F(MAINTLOG) DA(MAINT.LOG) MOD REU;
DATA _NULL_;
  SET MKDIR;
  FILE DOS;
  PUT 'MKDIR ' DIR;
  FILE MAINTLOG;
  PUT  'MKDIR ' DIR;
RUN;
TSO FREE F(DOS MAINTLOG);
TSO ALLOC F(DOS) DA(DOS.BAT) MOD REU;
TSO ALLOC F(MAINTLOG) DA(MAINT.LOG) MOD REU;
DATA _NULL_;
  SET RMDIR;
  FILE DOS;
  PUT 'RMDIR ' DIR;
  FILE MAINTLOG;
  PUT  'RMDIR ' DIR;
RUN;
PROC DOWNLOAD INFILE=DOS OUTFILE='D:\PCM\MAINT.BAT'; RUN;
/************************  STEP 1  *******************************/
/* Execute CLIST to free all the files                           */
/*****************************************************************/
TSO PCMEXIT;
```

CONCLUSION

Limitations

One limitation of the PCMAINT system is that it depends on a number of critical files to run (base SAS, terminal emulation software); if these basic files have been deleted or corrupted, the machine must be completely restored from backup diskettes. Secondly, the rate of file transfer makes it impractical for large-scale transfers.

One way to improve the transfer rate would be to keep compressed copies of the PC files on the mainframe; however, this would require an additional step to "uncompress," the files after downloading. It is also possible to build a check into the system to warn the user if the total number of bytes which need to be downloaded is excessive; at some point, it would be more efficient to restore the machine from backup diskettes. Finally, a useful enhancement to the system would be to allow the user to specify maintenance of only parts (that is, subdirectories) of the machine instead of the whole machine at once.

Summary

The PCMAINT system enables us to maintain the software on a number of remote PCs via a mainframe connection. When the system is run on a PC, a comparison is made between the files currently on the machine and a master list describing the files which should be on the machine. If any discrepancies occur, the PC is corrected by downloading copies of the correct PC files from the mainframe and erasing any extraneous files on the PC. This system has been useful in giving us control from a central location to assure the integrity of our remote training PCs and to keep them updated with the most recent software in a timely manner. It is also an efficient, automatic way to accomplish a task which had previously been tedious and time-consuming. The system provides many of the capabilities of networking for little of the cost; it's like using the mainframe as a network server.

I would like to thank Jerry Mock and Chris Williams of SAS Institute Inc., who assisted in the development and implementation of the PCMAINT system.