

Introducing the Sample Library File Contribution Server

Richard D. Langston, SAS Institute Inc. and
Gregory S. Nelson, University of Georgia

ABSTRACT

This paper discusses the repository of user-contributed SAS® applications software that will reside on an FTP file server accessible via Internet. This paper describes the various tools that allow for uploading and downloading SAS macro code, SAS programs, and SAS/TOOLKIT® modules.

INTRODUCTION

The file contribution server has been established to allow for the distribution of SAS applications. This server allows for uploads and downloads to take place via Internet email or by anonymous FTP. SAS applications can be any application that uses the SAS System in some capacity. These applications can consist of basic SAS code, SCL in catalogs, or SAS/TOOLKIT implementations.

This paper is divided into several sections. The first section discusses how to make contributions to the server via Internet email or anonymous FTP. The second section describes what happens during evaluation of your contribution. The third section describes how to download contributions via email or anonymous FTP. The next section discusses the notification service. The last section gives technical information on how the server is implemented, for those who are interested.

THE FILE CONTRIBUTION SERVER

The file contribution server is a program that resides on the userid `sascontrib@sasserv.uga.edu` at the University of Georgia. This program is activated by electronic mail messages. The subject of the email message indicates to the server what action is to take place. The possible subjects are:

- request contribution to file server
- cntrbxxx: ftp filename
- cntrbxxx: upload filename <m/n>
- cntrbxxx: download <filename>
- subscribe <daily/weekly/immediate>
- help
- index

where `cntrbxxx` is a contribution number and `filename` is a file to be uploaded or downloaded.

The details of the commands will be discussed in the subsequent sections of this paper.

The user can obtain help by sending email with the subject `help` and can get a list of existing contributions by sending email with the subject `index`.

MAKING CONTRIBUTIONS

We will assume that you have already prepared your application and that it has been properly tested by you and by your users. Once you're ready to contribute, send email to `sascontrib@sasserv.uga.edu` with the following subject:

request contribution to file server

The casing of the message does not matter, but this must be the exact wording.

The first line of text in your email message will be recognized as a one-line description of your contribution. All subsequent lines in this mail message will be ignored. It is not necessary to provide this description line, but it is highly encouraged so that an index of applications will show this one-line description. If you use a signature as part of your mail message, you must supply a one-line description so that the first line of the signature will not be confused with a description line.

You should receive an automated email response within 24 hours. The response will provide you with a contribution number. You should use that number in all subsequent email correspondence. Once you have this number, you can begin providing your contribution. You can provide your contribution either through email or through anonymous FTP, both of which are explained below.

You will need to send the following as part of your contribution:

- an index of all file names, in a file called `index`
- a very basic document, in a file called `readme`
- complete documentation in one or more files
- all SAS programs pertinent to the contribution

And if you are contributing binary data:

- all pertinent SAS data sets in transport format, converted to `tktdump` format
- all pertinent SAS catalogs in `cport` format, converted to `tktdump` format

See below for a discussion on the `tktdump` format.

And if you are contributing applications written using SAS/TOOLKIT Software:

- the source code
- any include files
- control file(s) for building the executable(s)
- instructions for building

The naming conventions to be used are as follows: all files should have all lowercased names, with the exception of the uppercased `.Z` used as a compression extension. Files will always have a two-level name, the first being up to 8 characters, the second being up to 3.

- `sas`: a SAS program
- `xpt`: TLKTDUMP file for transport data set
- `cpo`: TLKTDUMP file for CPORT version of catalog
- `doc`: documentation

And for SAS/TOOLKIT applications:

- `c`: C program
- `for`: FORTRAN program
- `txt`: INCLUDE file for PL/I or FORTRAN
- `h`: #include file for C
- `mac`: COPY or MACRO for IBM Assembly language
- `bid`: script to build the executable

Note that the index and readme files are contributed by you with

just those single-level names, but they will be renamed by the file contribution server to have the contribution number as an extension. For example:

- `readme.003`: README file for contribution 003
- `index.003`: INDEX file for contribution 003

The files are named this way so one who uses anonymous FTP can download several different *readme* and/or *index* files without having one version overwrite another. However, it should also be noted that different contribution directories can have the same file name, but with different contents. For example, the file `test.sas` may appear with `cntb0002` and `cntb0003` and be different files.

ANONYMOUS FTP METHOD OF CONTRIBUTION

If you have access to anonymous FTP, you can send the contribution using this method. You will need to bundle all your software together into a UNIX[®] tar file or a PKZIP[®] file. If you bundle into a tar file, you must compress the file using the *compress* or *gzip* command. If you use a PKZIP file, the file is already compressed. The file must be named one of the following (using `cntb0003` as an example contribution number):

- `cntb0003.Z` (tar file compress'd)
- `cntb0003.gz` (tar file gzip'd)
- `cntb0003.zip` (PKZIP file)

The anonymous FTP site is `ftp.uga.edu` and the directory for contributions is `/pub/sas/contrib/incoming`. You will contribute the single file by issuing the following commands (again using `cntb0003` as an example):

```
ftp ftp.uga.edu
user anonymous
(enter your fully-qualified userid as password)
cd /pub/sas/contrib/incoming
put cntb0003.Z
put cntb0003.gz
put cntb0003.zip
quit
```

(You will issue only one of the *put* commands, uploading whichever file you have created).

After completing the FTP, you should send a message to `sascontrib@sasserv.uga.edu` with the subject indicating that you have FTP'd the file. The contents of the message are ignored. For our examples above, the subjects would be:

- `cntb0003: ftp cntb0003.Z`
- `cntb0003: ftp cntb0003.gz`
- `cntb0003: ftp cntb0003.zip`

Once the message is received, the file will be processed by the file contribution server.

INDIVIDUAL FILE METHOD OF CONTRIBUTION

If you don't have access to anonymous FTP, you can email each component in a separate email message.

Your subject for each email should be as follows:

`cntbxxxx: upload filename m/n`

where *cntbxxxx* is your contribution number, *filename* is the filename being emailed, *m* is the part number and *n* is the total number of parts. You do not have to state 1/1 if there is only one

part being emailed; that will be assumed.

If a particular component consists of more than 1,000 text lines, please separate the component into multiple pieces of up to 998 lines each. Precede the first text line in each mail message with

`BEGIN-----Cut Here`

and follow the last text line with

`END-----Cut Here`

AFTER COMPLETING YOUR CONTRIBUTION

For either method of contribution, the file contribution server verifies that all components have been received. This is done by examining your index file. Once all files specified in the index have been received, you will be sent a confirmation method stating that all components have been received. At that time, the files will be examined.

Once the files have been examined, they will be added to the downloadable directories. See below for a discussion on the layout of these downloadable directories.

CONTRIBUTION EXAMPLE USING EMAIL

Suppose you have a file called *mystuff.sas*. It's a neat SAS program. That's the file you want to contribute.

In order to do so, you'll also need *mystuff.doc* with the user doc. You'll need a brief readme file called *readme*. And you'll need a file called *index* containing the following lines:

- `index`
- `readme`
- `mystuff.sas`
- `mystuff.doc`

Once you have those files, you're ready to contribute.

Send email to `sascontrib@sasserv.uga.edu` with the subject

request contribution to file server

You should have a single line in the message body with the thumbnail description of the contribution, for example:

This is my killer app for the SAS System

You will receive a response giving you the contribution number. Let's assume it is `cntb0010`.

You then send four separate email messages to `sascontrib@sasserv.uga.edu`. Each message will contain the text of the file referenced in the subject. The four subjects will be, respectively:

```
cntb0010: upload index
cntb0010: upload readme
cntb0010: upload mystuff.sas
cntb0010: upload mystuff.doc
```

Once all have been processed, you should receive a response stating that your contribution is complete.

Once the contribution has been evaluated, it will be placed in the downloading section of the anonymous FTP site. It will then be available for downloading via anonymous FTP or by the email server.

CONTRIBUTION EXAMPLE USING ANONYMOUS FTP

As in the section above, suppose you have *mystuff.sas*, *mystuff.doc*, *index*, and *readme*. Those are the files you want to contribute.

Send email to sascontrib@sasserv.uga.edu with the subject
request contribution to file server

You should have a single line in the message body with the thumbnail description of the contribution, for example:

This is my killer app for the SAS System

You will receive a response giving you the contribution number.
Let's assume it is *cntb0010*.

Assume the files of interest are the only ones in your current UNIX directory. You issue the following commands:

```
tar -cvhf /tmp/tar.dat *
compress /tmp/tar.dat
```

You now have a file called */tmp/tar.dat.Z* containing the compressed tar file. If you were on a PC, you would have entered the following commands:

```
pkzip cntb0010.zip *.*
```

and you would have a file called *CNTB0010.ZIP* containing the files to be contributed.

You would then invoke FTP and make an anonymous FTP connection. This is the probable command stream for UNIX:

```
ftp ftp.uga.edu
cd /pub/sas/contrib/incoming
put /tmp/tar.dat.Z cntb0010.Z
quit
```

After the FTP is completed, you send email with the subject

cntb0010: ftp cntb0010.Z

and the message body is ignored.

Once all have been processed, you should receive a response stating that your contribution is complete.

EVALUATION OF CONTRIBUTIONS

Once your contribution is complete, it will be evaluated. The purpose of this evaluation is simply to ensure that the application will work correctly when downloaded and tried by someone other than the developer. Also, issues of portability must be considered, and it might not be feasible for the developer to test the application on multiple platforms.

An attempt will be made to test the application on an IBM® mainframe, and a VAX/VMS® system using Release 6.08 of the SAS System, a UNIX system using Release 6.09, and a PC system using Release 6.10. Of course, if your application is supposed to work only on one given operating system, you can indicate this in your documentation.

Also, the evaluation process is used to compile and link SAS/TOOLKIT applications. In the interest of security and to ensure no licensed software is inadvertently linked, all executables must be linked during the evaluation process. This also avoids virus problems, since no "unknown" code is linked into

the executable.

Once the evaluation is complete, the entire application is moved to a download directory, where it is accessible via FTP and the email server. You are notified when the application is made available. Also notified are any individuals who are subscribed to the notification service. See the section NOTIFICATION SERVICE for more details.

DOWNLOADING CONTRIBUTIONS

The user can download via anonymous FTP or via the mail server. For FTP, things are very straightforward and similar to what one sees on other systems.

Choose a directory for downloading. If you are uncertain about when directory to choose, download the file *root/abstracts*, which contains the brief abstracts for all approved contributions.

Once you choose one of the contribution directories, you can download any and all files within the directory. The *readme* file contains a brief abstract, and the *index* file contains a file list. If the contribution is a SAS/TOOLKIT application, you can choose to download the source and script files to rebuild it for your host (if you are a licensed SAS/TOOLKIT Software site), or you can hope that the contribution directory contains a version of the executable program for you to download. The SAS/TOOLKIT application files are located in the subdirectory *toolkit* within the contribution directory. And within the *toolkit* directory is the source and script files, along with a subdirectory for each operating system for which a test link was made.

The executables are stored in a special text format called TLKTDUMP format. This format can be read by the SAS program TLKTLLOAD to convert into a local executable program for your operating system. (The TLKTLLOAD program can be found in *root/tlkload.sas*).

Also, any catalogs or transport datasets are stored in TLTKDUMP format. This means that there is no binary files stored in the directories.

The FTP address is *ftp.uga.edu*. Once connected, change the directory (via the *cd* command) to */pub/sas/contrib*. Within that directory, there will be a subdirectory for every approved contribution. Plus, the directory *root* will contain the necessary tools and info to use in conjunction with the items to download.

To download using email, you will send an email message to sascontrib@sasserv.uga.edu. Your email will activate a server program that will do different things based on the subject provided. Use the subject

cntbxxxx: download filename

where *cntbxxxx* is the contribution number, and *filename* is the name of the file to be downloaded. You will be sent the file in an email message. To get all the files in the *cntbxxxx* directory:

cntbxxxx: download

If no filename is given, it is assumed that all files associated with the contribution are to be downloaded. However, if a *toolkit* subdirectory is associated with the download directory, files will not be automatically sent from that directory. You will have to send a different email to get those files.

Each file will be sent in a separate email message. If any file is too large, it will be split into parts and sent in separate mail messages with part numbers in the subject.

DOWNLOAD EXAMPLE USING EMAIL

You want contribution 15, using the contribution number *cntb0015*. That contribution contains the files

```
index
readme
neatpgm.sas
neatpgm.doc
```

To get all the files for that contribution, send email to *sascontrib@sasserv.uga.edu* with the subject

```
cntb0015: download
```

Because no filename was specified, you will receive all four files in four separate email messages.

DOWNLOAD EXAMPLE USING ANONYMOUS FTP

As in the previous example, you want contribution 15. Use these commands:

```
ftp ftp.uga.edu
cd /pub/sas/contrib/cntb0015
mget *
quit
```

The *mget ** command, if available, will download all the files in the directory. If *mget* is not available, you can use the *get* command for each individual file.

NOTIFICATION SERVICE

Since there can be many applications being contributed, you may want to be informed when applications have been evaluated and made available for downloading. You can be kept informed of the contributions via the file server notification service.

You subscribe to the service by sending email to *sascontrib@sasserv.uga.edu*. The subject should be

```
subscribe xxx
```

where *xxx* is *daily*, *weekly*, or *immediate*. The message body is ignored. Once your mail message is received and processed, you will be notified of new contributions as per your request. Daily notification means that you are sent abstracts at 7:00 am Eastern Time after the corresponding contributions are accepted. Weekly notification means that you are sent abstracts at 7:00 am Eastern Time Wednesdays. Immediate notification means that you are sent the abstract whenever a contribution is accepted.

You can change your subscription at any time by resending the subscribe subject with a different level of daily, weekly, or immediate.

You can unsubscribe by sending email with the subject

```
unsubscribe
```

Note that only the userid used to send the email can be subscribed or unsubscribed.

EXAMPLES

- userid *abc* has IMMEDIATE subscription
- userid *def* has DAILY subscription
- userid *ghi* has WEEKLY subscription

Contribution *cntb1234* completes its evaluation and is accepted at 9:20 am Eastern Time Monday. *abc* is sent an email at 9:20 Eastern Time. *def* is sent email at 7:00 am Eastern Time Monday. *ghi* is sent email at 7:00 am Eastern Time Wednesday (two days later).

Contribution *cntb1235* completes its evaluation and is accepted at 2:50 pm Eastern Time Monday. *abc* is sent an email at 2:50 pm Eastern Time. *def* is sent email at 7:00 am Eastern Time Tuesday. *ghi* is sent email at 7:00 am Eastern Time Wednesday (two days later).

Contribution *cntb1236* completes its evaluation and is accepted at 2:50 pm Eastern Time Wednesday. *abc* is sent an email at 2:50 Eastern Time. *def* is sent email at 7:00 am Eastern Time Thursday. *ghi* is sent email at 7:00 am Eastern Time the following Wednesday (one week later).

Note that the immediate subscription causes one email to be sent for each completed contribution. The daily and weekly subscriptions cause one email message to be sent for all contributions completed since the previous email. An abstract is provided in each email for each contribution. The abstract corresponds to the *readme* text that was originally contributed.

TLKTDUMP AND TLKTLLOAD

These two SAS programs are responsible for converting binary data (transport files, CPORT files, executables) into a text stream that contains only printable characters. These programs have the builtin intelligence to properly handle the eccentricities of executable modules on MVS, VM/CMS®, and VAX/VMS platforms, as well as transport and CPORT files. Although the UNIX *uencode/udecode* commands have the same intention, these commands have several drawbacks: they are not standard commands in non-UNIX environments, they have no standard way of dealing with split files, and they have no validation that the data given to them is correct. The TLKTDUMP/TLKTLLOAD programs have all of these capabilities in place. They are portable across platforms, they have a builtin section delimiter recognition for split files, and they have checksumming for data validation.

For all contributions to the file server involving binary data, you must use TLKTDUMP. Since you cannot contribute executable programs, you will only need to use TLKTDUMP for transport files and CPORT files (when downloading catalogs). However, you may need TLKTLLOAD any time that you download contributions. This is because executables are prepared during the evaluation process and are put in place for downloading using TLKTDUMP. You will not need TLKTDUMP for applications that do not contain any binary data.

TLKTLLOAD is available via the filename *tools/tlktload.sas* and TLKTDUMP is available via the filename *tools/tlktdump.sas*. Both SAS programs have documentation in the headers to fully describe how to use them, but they are briefly explained here.

Both SAS programs consist of macro definitions. These macros are as follows:

```
%TLKTDUMP(LRECL=lrecl,
          INFILE=input-library,
          OUTFILE=dump-file,
          INTERVAL=byte-count,
          USERINFO=userinfo-fileref,
          EXPORT=YES | NO,
          IEBCOPY=YES | NO);
```

The *LRECL=* specifies the maximum number of bytes to place in

a single record in a dump file. Its default is 72. It is recommended not to exceed 72 because some electronic mail systems start wrapping lines longer than 72.

The **INFILE=** option refers to the file to be dumped. This reference can be a fileref or a quoted string to a file name. Note that load modules on MVS, the file refers to the entire load module library, and all modules in the library will be dumped. On VM/CMS, the file refers to the entire LOADLIB file, all of which will be dumped. All other platforms refer to a single binary file to be dumped. The **INFILE=** option is required.

The **OUTFILE=** option refers to the output dump file. This will be either a fileref or a quoted string giving the name of the output file. The **OUTFILE=** option is required.

The **INTERVAL=** option is the byte count that TLKTDUMP will use in determining the number of parts of the file. Each part is delimited by a special "part record" (explained below). The entire dump goes into one file, but the individual parts can then be later extracted into different files for separate file transfer. This is convenient for electronic mail systems that have byte count limitations. Note that the **INTERVAL=** option doesn't operate on the exact byte count, and that there is no way for TLKTDUMP to know the number of header bytes that your mail subsystem may insert when mailing a file. Therefore, you should reduce **INTERVAL=** by about 1000 bytes from your desired interval amount. The default value of **INTERVAL=** is -1, meaning there will be only two parts, part 0 (containing header info), and part 1 (containing the rest of the dump).

The **USERINFO=** option is a fileref or quoted string indicate the file name for a data file containing additional information to be placed in the dump header file. This file can be any length and contain any data desired (although there should be no binary data in the file that would introduce difficulties in file transfer). The default is to have no userinfo data in the header.

On MVS, the **IEBCOPY=** option should be used for dumping load modules. The option will cause additional information to be saved in the dump header so that an IEBCOPY dump file can be successfully recreated.

The **EXPORT=** option indicates that the binary file being dumped is from the transport engine or from PROC CPORT, and is not an executable program. This option is used so that this information can be used when recreating the binary file with the proper attributes.

You will use the TLKTDUMP program to produce a text-oriented, non-binary stream that can be emailed successfully. This should also be the format used for your CPORT and TRANSPORT files that are being uploaded via anonymous FTP.

If you are downloading a TLKTDUMP file, you can use the TLKTLLOAD program to convert the file back into the appropriate binary stream. The syntax of TLKTLLOAD is much simpler than TLKTDUMP:

```
%TLKTLLOAD(IN=infile,OUT=outfile);
```

where *infile* is a fileref that refers to the TLKTDUMP file and *outfile* is a fileref that refers to the resultant binary file. Note that with load modules on MVS, the *outfile* fileref refers to a temporary file to be passed to IEBCOPY in a later jobstep.

EXAMPLES OF TLKTDUMP AND TLKTLLOAD

If you have downloaded a TLKTDUMP file that contains a load

module for use on MVS, you can use the following JCL to convert the dump file into a load module:

```
//          EXEC SAS
//UNLOAD2   DD DSN=%%UNLOAD2, DISP=(NEW, PASS),
//          UNIT=SYSDA, SPACE=(TRK, (50, 50))
//DUMPFILE   DD DSN=<my.dump.file>, DISP=SHR
//SYSIN      DD DSN=<tlktload.program>, DISP=SHR
//          DD *
%TLKTLLOAD(INFILE=DUMPFILE,OUTFILE=UNLOAD2);
//          EXEC PGM=IEBCOPY
//IN         DD DSN=%%UNLOAD2, DISP=SHR
//OUT        DD DSN=<my.load.library>, DISP=SHR
//SYSUT1     DD UNIT=SYSDA, SPACE=(TRK, 50)
//SYSUT2     DD UNIT=SYSDA, SPACE=(TRK, 50)
//SYSPRINT   DD SYSOUT=A
//SYSIN      DD *
COPY INDD=IN,OUTDD=OUT
//
```

where *my.dump.file* is the file containing the downloaded TLKTDUMP file, *tlktload.program* is the TLKTLLOAD program downloaded from the *tools* directory, and *my.load.library* is the name of the load module library to contain the new module.

If you have an TLKTDUMP file that contains the dump of an executable image for a specific UNIX system, you can run the following shell script on that UNIX system:

```
cat > readdump.sas <</
%inc 'tlktload.sas';
%tlktload(infile='dump.dat',outfile=mymod);
/
sas readdump.sas
```

where *tlktload.sas* is the file downloaded from the *tools* directory, *dump.dat* is the file containing the downloaded TLKTDUMP file, and *mymod* is the name of the executable to be created.

If you have a transport file to be converted into a TLKTDUMP file, here is the sample code for any UNIX platform:

```
cat > writdump.sas <</
%inc 'tlktdump.sas';
filename userinfo 'userinfo.dat';
data _null_; file userinfo;
    put 'this is my test file';
run;
libname testfile xport 'testfile.dat';
* create a sample transport file;
data testfile.temp; x=1; run;
%TLKTDUMP(LRECL=72,
          INFILE=testfile,
          OUTFILE='testdump.dmp',
          INTERVAL=50000,
          USERINFO=userinfo,
          EXPORT=YES,
          IEBCOPY=NO);
/
sas writdump.sas
```

where *testfile.dat* is the transport file being created and then read from, and *testdump.xpt* is the TLKTDUMP file being created.

HOW THE MAIL SERVER WORKS

The mail server functions by using the *filter* component of the UNIX *elm* mailer. The *filter-rules* file contains conditions for checking for subjects containing *request contribution to file server*

and for subjects containing *cnfb*. Note that *filter* does its comparisons without case-sensitivity. If a proper subject is recognized, the filter invokes the file server SAS program.

The SAS program reads the subject to determine the action to take. For contributions contained in the email text, *stdin* is read. For contributions contained on the FTP site, the data are read directly from the FTP site.

The SAS program maintains email address lists for subscription info, informs contributions about their contributions being complete, and informs subscribers of status changes. In short, it takes care of all the data management functions.

CONCLUSION

The file contribution server will provide a repository for SAS users to make contributions of their SAS programs. This server will be available to anyone with Internet electronic mail access, not just for those with anonymous FTP access.

DISCLAIMER

This file server is still under development at the time of this writing, and some aspects are certainly subject to change. It is recommended that you subscribe using the process described in the NOTIFICATION SERVICE section above. Once subscribed, you will be notified when the file server is functional and how to obtain current documentation.

REFERENCES

SAS and SAS/TOOLKIT are registered trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

MVS and VM/CMS are registered trademarks of International Business Machines.

VAX/VMS is a registered trademark of Digital Equipment Corporation.

UNIX is a registered trademark of Novell, Inc. X/Open Company, Ltd., is the exclusive licensor of such trademark.

PKZIP is a registered trademark of PKWare, Inc.