Paper 182-25

The SAS® System Electronic Mail Interface: A Case Study

Derek Morgan, Washington University Medical School, St. Louis, MO

Abstract

The 1999 International Genetic Epidemiology Society meeting used software developed with the SAS System for registration and abstract management. Given the international nature of the presenters and attendees, however, conducting the routine business associated with these tasks by telephone or postal service would quickly get expensive. To reduce administrative costs, the organizers wanted to use electronic mail as the primary approach for confirming registration and abstract submission. It was also the preferred method of notifying presenters of their acceptance, and to give them the time and date of their presentation.

The SAS System electronic mail interface not only fulfilled these requirements, it did so in a seamless, integrated fashion requiring no additional training or user intervention.

Why IGES '99 Chose The SAS System

The original decision to use the SAS System for registration and abstract management was based on the existence of similar in-house software. This software had been developed for the 1998 MidWest SAS Users Group conference. Registration confirmation in the original software was handled via printed letters mailed to registrants. The letters were produced using a macro library that generated PostScript code[1], which was then sent to a network printer. The resulting letters were folded, placed in labeled envelopes, stamped using a postage machine, and mailed first-class.

The regional nature of the MWSUG conference allowed for accurate budgeting of postage costs. However, IGES projected many attendees and presenters from overseas, adding airmail costs, the special handling of overseas envelopes, and extra aggravation for the conference coordinator.

Electronic mail was clearly advantageous in terms of cost, but how would it fare in the convenience category? The administrative applications were to be built on our UNIX server using version 6.12 of the SAS System, so there would be no problem accessing the standard e-mail server, keeping development time minimal. It was also completely transparent to any user, with the only possible indication of an electronic mail link being the email address field on the data entry screen.

How the Application Works

A previous SAS System application requiring automated electronic mail had been developed several years earlier. It utilized UNIX piping and required some set-up work with the UNIX mail program to provide aliases for five fixed distribution lists. This method, while feasible, was not well suited for this task, since the email addresses would not be known in advance for the creation of an alias file.

The SAS System electronic mail interface allows the recipient's address to be read from a column in a table. From there, a customized electronic letter can be generated in a DATA step or through SAS Component Language code, and sent with a subject line that may also use values in a table. The mail can also be copied to other addresses. Attachments can be included if desired.

For this particular application, a DATA step was chosen as the medium for generating the letter. SCL could have been used to create the letter from within the FSEDIT application itself. However, it was decided that maintenance in open code was much easier. In addition, simple text changes could have been effected by the conference coordinator without programmer intervention. Attachments were not used since some electronic mail systems either can not or will not handle attachments. The registration confirmation electronic mail contained information from the registration database such as: the registration number, payment status, events registered for, and standard text with general conference information. A second e-mail report was also generated containing a summary of the transactions for the session. This report was sent to the conference coordinator and the programmer in charge of maintenance.

The front end of this application is a custom FSEDIT screen. The entire application is invoked from a UNIX command line with a single word (through aliasing). The software is accessed using an X terminal running on Windows NT clients, allowing use of the software from any machine with the correct permission. Although there were separate commands for registration and abstract management, it would have been a simple matter to tie the application together using SAS/AF® to create an integrated, menu-driven conference management system.

The SAS System Electronic Mail Interface

Adding electronic mail capability to SAS System applications is a matter of creating the text of your mail and using the e-mail tokens to address, give the message a subject, and send the mail. There is also a token to send a copy of the mail to other recipients. In this application, the mail was created using PUT statements inside of a DATA _NULL_ step. It's old technology, but easy to modify and debug. This method also has the advantage of being separate from the editing of the tables from which it obtains information.

You need to start the SAS System with the -EMAILSYS system option. This cannot be set inside the program, but must be defined at invocation of the session. What the parameter for the option will be is operating system-dependent. For UNIX, there are two possibilities: one that allows for aliasing and attachments, and one that does not.

Once it has been started, define a FILENAME with the EMAIL option as below:

```
FILENAME mailout EMAIL 'derek@wubios.wustl.edu';
```

This sets up the file reference where the output is to be directed and gives it a default address. There are multiple ways to specify the actual address to be used for the mail. We used the e-mail token "!EM_TO!" inside the DATA step that created the text of the mail. In this DATA step, first we direct the output to the file reference we specified earlier:

```
FILE mailout NOPRINT;
```

Next, we address our mail, give it a subject, and specify the address where a copy of the mail is to be sent.

```
PUT '!EM_TO!' email;
PUT '!EM_SUBJECT! Registration Confirmation #'
        regno z3.;
PUT '!EM_CC! derek@wubios.wustl.edu';
```

The address is contained in the table variable “EMAIL”. Note that the e-mail token and fixed text for the subject line are within the same set of quotes. A registration number ("REGNO"), generated by the FSEDIT application, is attached to the subject line. Since the CC: address doesn't change with each record, it is hard-coded in the "!EM_CC!" token.

After the mail address and subject have been set up, PUT statements provide the means for the e-mail text. The DATA step allows for customization based on table values as demonstrated by the following example:

```
PUT 'You have registered for:';
IF regtype GT 0 THEN
    PUT 'IGES Conference: ' regtype regtyp.;
IF morton THEN
    PUT 'Morton Symposium';
IF banquet1 GT 0 THEN
    PUT @1 banquet1 2. ' extra banquet tickets';
```

In this example, each item is on a separate line. Use of a trailing "@" with the PUT statement would allow for the "flowing" of text.

After the statements creating the text for the message have been coded, use the token "!EM_SEND!" to actually send the message at the end of the record. Without this token, only one message would be sent using the data from the last record in the table. Following that, you need to clear the previous message and create a new one for sending to the next person in the table. The "!EM_NEWMSG!" token does that, clearing the recipient and copy to lists, the subject line, and text of the message. A special token, "!EM_ABORT!" is used to override the default behavior of the electronic mail interface. Normally, the message is automatically sent at the end of the DATA step. Using this token will prevent that from happening when there are no more records to be processed. Otherwise, a blank final message (because of the EM_NEWMSG token) would be transmitted to any addressees that are hard coded in the TO and CC tokens. The EM_ABORT token must be used in conjunction with the EOF option on the SET statement.

```
/* Send message and clear for next message */
DATA _NULL_;
SET maillist EOF=last;     ①
        …
PUT '!EM_SEND!';
PUT '!EM_NEWMSG!';
RETURN;

/* Don’t send at end of DATA step */
last:  PUT '!EM_ABORT!';
RUN;
```

The EOF option on the SET statement (①) is necessary for the proper operation of the EM_ABORT token. Only the very last send operation should be aborted, which occurs during the last record. Therefore, the EM_ABORT token should only be executed during the last record. This prevents the default send operation from occuring.

Debugging the e-mail text and/or the process is usually simple, most frequently involving a missing exclamation point for one of the tokens, or that perennial SAS System favorite, mismatched quotes.

As mentioned earlier, a second e-mail report is generated using a separate DATA step. Instead of one e-mail per record, however, we want one e-mail for the whole table. In this case, we can dispense with the "!EM_SEND!", "!EM_NEWMSG!", and "!EM_ABORT!" tokens, relying on the default of sending the e-mail message at the end of the DATA step.

```
DATA _NULL_;
SET tosend;
BY DESCENDING econfirm;
FILE mailout NOPRINT;
today = TODAY();
PUT '!EM_TO! derek@wubios.wustl.edu';
PUT '!EM_CC! techprog@wubios.wustl.edu';
PUT '!EM_SUBJECT! Conference Confirmation
        Summary ' today date9.;
IF first.econfirm THEN DO;
    PUT ' ';
    PUT "The following registrations were " econfirm
            econfrm.;
    PUT "Reg #" @8 "Name";
END;
PUT @1 regno z3. @8 name;
RUN;
```

With the exception of the address, copy, and subject lines, this is nothing more than a simple DATA step report. This is why adding e-mail capacity is easy.

There is one remaining e-mail token that has not been discussed here. That is the "!EM_ATTACH!" token, which is used for sending MIME attachments with the mail. As with the other tokens, it can be executed conditionally, allowing different people to receive differing attachments. However, if you are going to use this, you must be positive that recipients are able to receive attachments. If not, this will be wasted effort.

Using Electronic Mail via SAS Component Language

Although the front end of the IGES applications used SAS/FSP and SCL, we separated the editing and entry of information from the e-mail process. It is possible to

perform the mail process from within the application using SCL. The only difference between DATA step and SCL generation of electronic mail is that the DATA step uses PUT statements to send text to the external file. In SCL, FPUT and FWRITE are what sends the text to the external file. A default filename still has to be created, and the tokens still have to be used in the same manner. Which method you use literally depends on your preference.

Problems With the Application

The primary problems were those people who had no electronic mail address, and incorrect addresses in the database. The former was handled via surface or air mail. If the e-mail address field was empty, the record was put in a different dataset. The PostScript macro library was then used from within the application to generate a PostScript file of letters for printing. The file was automatically sent to our PostScript printers by using the "X" command to invoke the UNIX print command from the SAS System program. The confirmation report e-mail notified the conference coordinator if there were letters waiting to be mailed.

The latter posed a problem, especially given the nature of our UNIX mail system set-up. The offended mail system would send incorrectly addressed e-mail back to the sending account, which was not the conference coordinator's account. For security reasons, using a .forward file was not possible, so the returned mail sat in the incoming mail folder on the sending account. The technical programmer was designated to check that queue and forward any undeliverable mail to the conference coordinator. The summary report was used to signal the programmer when this task was to be performed. So, there was a mechanism to handle incorrectly addressed mail, but it was not automated.

At the recipient's end, line length within messages could be a problem. Although many electronic mail systems intelligently wrap words, some systems have restrictions on line length. This issue was avoided in the current application by ensuring that lines did not exceed eighty characters.

The Future of Electronic Mail and The SAS System

With the presence of the Output Delivery System in version 7 and beyond, it is going to become even easier to generate output in a myriad of formats. As more mailers are able to handle attachments, mailing finished reports directly from applications will become more of a standard. In addition to web distribution, it will be possible to send pieces for inclusion as word processing documents, or even as finished PostScript files for printing by the recipient. One improvement to the e-mail interface that I would like to see is the addition of an "EM_REPLY_TO" token. This would make it easier to have processing and mailing done from a central location, while directing any replies (or incorrectly addressed mail) to a customer service representative.

Summary

The ability to link the SAS System with electronic mail streamlined the administrative process for the 1999 IGES meeting. The simplicity of the tools used made development fast and quite easy. By eliminating the need for user intervention in sending the e-mail, training needs were almost eliminated. Composing and addressing the mail based on table values allowed for a high degree of customization, and the entire operation was transparent to the user.

Although the process did not work for all cases, direct intervention by a programmer or user was kept to a minimum. While a REPLY_TO token would have been useful, it was not a major obstacle to function or acceptance.

This method holds promise for automating registration confirmation and conference administration, especially as a part of an integrated SAS System application. This integrated application could easily support web-based registration, as well as centralized data entry at the conference administration location.

Further inquiries are welcome to:

Derek Morgan
Division of Biostatistics
Washington University Medical School
Box 8067, 660 S. Euclid Ave.
St. Louis, MO 63110
Phone: (314) 362-3685 FAX: (314) 362-2693
E-mail: derek@wubios.wustl.edu

This and other SAS System examples and papers can be found on the World Wide Web at:

http://www.biostat.wustl.edu/~derek/sasindex.html

---

[1] Morgan, D. and Province, M. (1992). "A PostScript™ Macro Library," *Proceedings of the Seventeenth Annual SAS® Users Group International Conference*. Cary, NC: SAS Institute, Inc.