A computerized system for creating and maintaining Data Entry Instructions has been developed in SAS®, using SAS FSCALC and SAS/AF.

**Background**

At Stuart Pharmaceuticals, data from clinical trial Case Report Forms are computerized into SAS data sets from 80-column TSO records. These TSO records are punched by Corporate Data Entry according to a set of instructions provided by the Clinical Systems Analyst. Manual techniques available for the creation of data entry instructions include:

1. Writing directly onto a Data Entry Instruction form
2. Typing directly onto a Data Entry Instruction form
3. Storing in and printing out from a Word Processing document
4. Storing in and printing out from a TSO file.

This paper describes a new computerized system for creating and maintaining data entry instructions. Included in the discussion are the following topics: rationale for developing such a system, description of the system, and benefits of the system.

**Rationale**

The computerized data entry instruction system was developed in conjunction with a new drug project in which case report forms for many protocols differed in format from each other, but were not necessarily different in content, as illustrated by Figure 1. It was decided to call each 80-column collection of data a "module", and to develop separate data entry instructions for each module. It was anticipated that numerous changes in data entry instructions (e.g., adding new variables, changing lengths, etc.) would be required, and that a computerized system would be best suited to handle these changes accurately and efficiently.

**Outline of the System**

The Data Entry Instruction System consists of the following basic components:

1. SAS FSCALC screen and program for entering data and calculating column allocations.
2. SAS data set "D.MODULES" for retrieving and storing Data Entry Instruction information.
3. SAS/AF menu-driven facility which serves as a link between components (1) and (2) in order to execute the FSCALC program for a specified module name, and then store the resulting values in the "D.MODULES" data set.
4. Ability to print Data Entry Instructions from the "D.MODULES" data set.
5. Ability to create SAS INPUT statements from the "D.MODULES" data set.

**Details of the System**

1. **FSCALC Screen and Program**

   Figure 2 illustrates the FSCALC screen used for entry of data entry instruction information. The user fills in entries in the columns titled "NAME", "DESCRIPTION", "TYPE", and "COLUMNS". When ready to calculate the column allocations, the last row name is changed to "LAST", and "RUN TEST.PGM" is typed into the command line. At this point, the user can continue to modify column allocations as needed. Once satisfied, the user can return to DEFINE mode to change the row named "LAST" to the appropriate ITEM number name, and can then save the data entry information with an output statement: "OUTPUT COL D.NEW". To end, type "CANCEL".

   Figure 3 illustrates the FSCALC program used for calculating column allocations.

2. **SAS Data Set "D.MODULES"**

   The SAS data set which contains the result of the SAS FSCALC screen entries is called "D.MODULES". Its variables are listed in Figure 4. When a user specifies a module name to be processed, data set "D.MODULES" is searched for records with variable MODULE having the specified value. If found, the FSCALC screen will be filled with the pre-existing values of that module. If not found, a new screen will appear with only the first 3 items filled in. (These first 3 items are standard, and come from a "blank" data set called "D.MODEL").

3. **SAS/AF**

   The program that provides the link between the SAS FSCALC component and the SAS dataset "D.MODULES" is a member of the SAS/AF catalog "D.KPINSTR". Figure 5 illustrates the menu displayed as a result of executing the command "PROC DISPLAY CATALOG=D.KPINSTR.TEST.MENU". If the user chooses option 1 to process data entry instructions, the screen shown in Figure 6 is displayed; the user can then specify either a new module name for creating new data
entry instructions, an old module name for modifying an existing set of data entry instructions, or a blank to return to the main menu screen. The program statements which are executed in conjunction with the selection of option 1 are given in Figure 7. During the course of a data entry instruction session, all the modules which are accessed are given a value '* ' for the variable FLAG; this identifies all the modules for which print-out's or INPUT's may be requested later. In order to quit, the user returns to the main menu screen and either selects option 2 or presses the END key; if option 2 is chosen, three programs are automatically submitted to produce print-out's and create SAS INPUT statements for the flagged modules.

4) Printing of Data Entry Instructions
Data Entry Instructions can be printed in either of two ways: by module name, or by case report form number. Each module name is unique. The program to print out data entry instructions by module is shown in Figure 8, and a sample print-out is shown in Figure 9.

The program to print instructions by case report form number is a little more complicated because each module can be associated with more than one case report form, as mentioned earlier. (Refer to Figure 1.) The SAS data set "D.FORMS" describes the modules used in each case report form. The program to print data entry instructions by case report form is shown in Figure 10, and a sample print-out is shown in Figure 11.

5) Creation of SAS INPUT Statements
Information concerning variable names, types, lengths and column locations from data set "D.MODULES" can be used to create SAS INPUT statements to be used in the SAS programs for computerizing the TSO input data into SAS. Figure 12 gives an example of the program to accomplish this, and Figure 13 shows an example of the resulting INPUT statements.

Figure 14 diagrams the flow of information among the various components of the Data Entry Instruction system.

Benefits
Advantages of the Computerized Data Entry Instruction System are:

1) It is easy to develop data entry instructions directly on-line, modifying the column allocations until the count is satisfactory. This eliminates the need for scratch paper drafts. In addition, accuracy in the calculation of column locations is assured, whereas for all the manual methods there exists the possibility for miscalculation.

2) The data entry instructions which are stored in "D.MODULES" can be readily retrieved and modified.

3) The FLAG variable to identify newly created or newly modified modules in the "D.MODULES" data set is an important feature. New print-out's and new INPUT statements need only be created for those modules which are flagged.

4) The submission of jobs to produce print-outs and create INPUT's for the flagged modules can be done automatically as part of the SAS/AF component.

5) The use of the "D.MODULES" data set in the creation of the SAS INPUT statements is a powerful extension of this basic system: this feature automatically eliminates one of the steps necessary in the computerization process, and also ensures accuracy in the column specifications.

SAS is a registered trademark and SAS/AF is a trademark of SAS Institute, Inc., Cary, NC, USA.

Contact Author
Ruth F. Quah
Stuart Pharmaceuticals, Division of ICI Americas, Inc.
Wilmington, DE 19897
(302) 575-2235
ITEM1.FROM = 1; ITEM1.TO = ITEM1.COLS;
DO I = ITEM2 TO ITEM1.LAST;
    ITEM[I] = ITEM[I-1] + 1;
END;

FIGURE 3
PSOALC PROGRAM

CREATION OF DATA ENTRY INSTRUCTIONS
WHAT DO YOU WANT TO DO?
1 = CREATE/MODIFY DATA ENTRY INSTRUCTIONS
2 = QUIT

FIGURE 5
SAS/AF MENU

- 4M
PROC DISPLAY CATALOG=D.KPINSTR.TEST.MENU; RUN;
DATA TEMP; SET D.MODULES;
IF PSSOALC NAME='A'; IF ITEM NO NOT LAST;
DATA TEMP; MERGE D.MODULES TEMPSQALC;
DROP FROM TO MODULE;
DATA D.MODULES; SET D.MODULES;
PROC PSOALC CATALOG=D.KPINSTR.TEST.CALC DATA='fDoiP NODIR:
VAR NAME DESCRIBE TYPE COLS;
ID ITEH=10;
RUN;
DATA NEW; SET NEW END=EOF; IF EOF=1;
CALL SYMPUT( 'LAST'. N ); RUN;
DATA NEW; SET NEW; IF NLAST LE 3 THEN MODULE='CANCEL'; RUN;
DATA D.MODULES; SET D.MODULES NEW;
BY MODchar MODNUM ITEMNUM;
IF FIRST . ITEMNUM;
IF MODULE HE 'CANCEL'; RUN;
DATA NEW; SET NEW(RENAME=(ITEMNO ROW) I);
DROP MODCHAR MODNUM ITEMNUM;
PROC DISPLAY CATALOG=D.KPINSTR.A12.PROGRAM: RUN;
...

FIGURE 7
SAS/AF PROGRAM

287
**Fig 5**

**Fig 6**

**Fig 7**

**Fig 4**

**Fig 2**

**Fig 3**

**PROC DISPLAY CATALOG=**

**D.KFINSTR.TEST.MENU**

**Start**

**Option 1**

**Menu:**

1 = Process
2 = Quit

**Option 2**

Submit Jobs
Fig 8, 10, 12

End

**Blank**

**Module**

**Name**

Output
Fig 9, 11, 13

**Specify**

**Module**

**Name**

**Non-Blank**

**Module**

**Name**

**AF Program to:**
1. retrieve MODULE values from D.MODULES
2. load MODULE values into FSCALC screen
3. store results of FSCALC back into D.MODULES
4. return to MODULE specification screen

**Fig 14**
SYSTEM FLOW