Paper 90

Automated Input Interface for SAS® Batch-Mode Processing

S. Patrick Thornton, University of California San Francisco, San Francisco, CA

ABSTRACT

The Automated Input Interface (AII) was created as a tool for the intermediate base SAS® software user in order to facilitate batch processing of raw data through the specification of a simple set of parameters, rather than through the direct creation of syntax. A special data set, termed the AII control data set, is used in conjunction with SAS System macro code to produce the syntax necessary to create system data sets from raw data. The AII control data set determines the data set names, variable names, and variable types for the conversion of raw data sets to SAS System data sets. Thus, all data sets that are created with the AII and that exist in multiple projects are consistent and can readily be included in automated restructuring, merging, analyzing, routing, and reporting. AII offers the greatest advantage when many projects are dependent on large data sources which are stored in multiple record structures and physical locations (e.g. tapes). The input of any number of new raw data records can be achieved through the one time addition of information to an AII control data set. AII also handles the repetition of input syntax, and the creation of any combination of multiple data sets from within one DATA step.

INTRODUCTION

AII was created to run on a MVS operating system using TSO. At this location, SAS software was only licensed for batch execution, and Job Control Language was required for program execution. The data sources were sequentially stacked VSAM records that were stored on multiple tapes. Most SAS System projects used data stored in 1 to 5 data sources where most data sources had multiple records and sub-records that dictated the variables in each record structure. Most data sources were stored across 1 to 5 tapes. Multiple data sets were created from most DATA steps, such that, variables were inputted according to the value of record and sub-record indicators. Since the data sources were stored on multiple tapes, DATA steps were repeated for each tape source. The goal of AII was to automate the input process, produce standard data set names, variable names, and variable types across all projects, and allow the creation of data sets from new raw data sources without the need for the direct creation or alteration of SAS System syntax.

USING THE AII

An example configuration of the AII is found in Figure 1. Several sections in the program are designated as "change sections" where you will likely make changes in order to properly execute the program. Change section 1 defines the raw data sources that you want to convert to SAS System data sets. In this example the data sources are text files on a PC hard drive, however in the actual application these physical locations were on tape or disks drives on a mainframe operating system.

Change section 2 contains the AII control data set. The control data set is similar to a code table or data dictionary, but it has the added function of controlling, in conjunction with SAS System macro code, the conversion of the raw data to SAS System data sets. You must specify the data source (tap), record (recor), sub-record (subrec), variable name (varn), variable location (varl), format indicator (c), data access format (daf), and variable role (reci). The data source variable value must be the same as the first two characters of the macro variables that resolve to the corresponding file path in change section 1 (e.g. CM). The values of the record and sub-record variables must be the literal values of the record or sub-record indicators found in the raw data files (see limitations). The variable name values must specify the literal name for the variables to be created and must conform to SAS System variable name conventions. The variable location values must specify the location of the data in the corresponding data source by using the appropriate SAS System syntax for numeric, character, or date column INPUT. The format indicator variable values must be equal to 2 if the variable is a date and 1 if not. If the variable is a date then the data access format variable value must be set to the appropriate SAS System INPUT date format, else an "H" is used as a placeholder.

The AII also controls the role that each variable in the to-be-created SAS System data set(s) will play in the structure of the DATA step(s). The role variable (reci) can take the value of "R," "S," "P," or "H." The value of "R" indicates that the variable (value of varn) will be an indicator of record type. The value of "S" indicates that the variable will be an indicator of the sub record (see limitations). The value of "P" indicates that the variable is primary or common to all records in a data source (these are akin to key variables in a relational database--R and S are by default also primary). The value of "H" indicates that the variable is found only in the specific record and sub record in the data source.

To review, the AII control data set in Figure 1 is configured to represent the following. The "CM" data source has two record types, "AB and LQ", where each has a single sub record of "0." The record "AB," sub record "0," has 8 variables, one has a date value (offdat), two have character values (off, rty) and five have numeric values. The variable "RTY" is the designated record indicator and "CCN" is the sub record indicator. Every record in "CM" has a variable called "CDI" and "CAS" (primaries), and the variables "SPN, OFF, OFFDAT, and SNU" are unique to the "AB0" record structure.

In the actual application of the AII, the control data set is over 800 observations, and is stored in a SAS library rather than following a SAS System CARDS statement. The control data set is entered one time, and aside from corrections during beta testing it does not need to be altered except to append information regarding new data sources.

Change section 3 allows you to specify which data sources and records to convert from raw to SAS System data sets. The values of the variables in the data set "first" must match precisely with the values of corresponding variables found in the control data set. The data source variable (t) and the record variables (r1-r10) must contain values found in the corresponding variable values of the control data set, with the exception that the record parameter values also contain a sub-record value if one exists (e.g. AB0 where AB indicates record and 0 indicates sub-record). The data set "first" may contain any number of observations where each observation is a request for the creation of a data set(s) from variables found in specific records and sub-records from a single data source. The number of physical locations (nt) for a data source of the exact same structure must also be specified. The value the data source (t) and the number of data sources (nt) are concatenated by the macro code to form a macro variable (e.g. CM1) that resolves to the identification of an INFILE. The number of data sources (nt) cannot be greater than the number of physical locations for a given data source as specified in change section 1.

Change sections 4 and 5 are optional and allow you to select only certain variables from the data sources and records that you chose in change section 3. Change section 4 requires that you specify whether a selection of variables is requested. Macro variables "sel1-sel3" correspond to the first through the third data source (i.e. 1st-3rd observations in the data set "first"). When the program in Figure 1 executes, it creates SAS System syntax necessary to create SAS System data set(s) from the data source specified in the first observation in data set "first." If the

marco variable "sel1" resolves to the value "yes" it will cause only the matching user-selected variables in section 5 to be included in the SAS System data sets created for the corresponding data source. Thus, all variables for all records in the data source that you want to include must be in section 5 if the corresponding selection macro variable resolves to "yes." When the selection macro variables are blank, all variables from a particular data source that are in the AII control data set will be part of the SAS System data sets that are created. Like change section 3, section 5 requires that the value of the variables correspond precisely with the values of like variables in the control data set. The number of variables (e.g. v1-v15) that you may enter from a given data source, record, and sub record is dynamic (e.g. V1-Vn).

AII OUTPUT

AII creates SAS System data sets from raw data, standardizes data set and variable names, and appends data sets of like structure that are created from more than one physical location. To illustrate, part of the execution of the AII configuration in Figure 1 is shown in Figure 2. Four intermediate data sets "CMAB01, CMAB02, CMLQ11, and CMLQ12" were created where each is named as a sequential concatenation of the data source (e.g. CM), record (e.g. AB), sub record (e.g. 0), and data source number (e.g. 1). Each DATA step created two system data sets. The number of data sets created was controlled by the value of number of records variable (nr) in the data set "first," and the record types by the values of "r1," and "r2." Two DATA steps were executed, one for each physical location of the data source "CM" as shown in change section 1 and controlled by the variable "nt" in data set "first." Eight "AB0" variables and seven "LQ0" variables were included in the KEEP statements. The number of variables was dictated by the number of observations of each record sub record type in the control data set. The first INPUT statement in the DATA steps included all variables with a variable role indicator of "R, S, or P," thus meaning that the variables are common to all records in the "CM" data source. The conditional IF statements included only variables with role indicators of "R or S," and the INPUT statements enclosed by the IF statements contained only variables with a role indicator of "H." The later indicates that these variables are unique to the record and sub record type.

The AII used PROC APPEND to create "ACMAB01" and "ACMLQ11" where all variables have standard names and types. The standardization can have tremendous effects on efficiency. For example, if 50 SAS programs were built over a 4 year period by several different programmers to produce monthly reports, the programs may rely on the same 12 records obtained from 6 data sources, yet the programs may not have standard variable names and types. Thus, monthly reporting may require that the 50 programs execute from consecutive input processes, have repetitive access of the same data sources, and produce data sets that are incompatible for merging. If AII were used to handle input during the development of the 50 programs then all programs could obtain data from a single data access process, and automation could be planned to capitalize on the standardization.

LIMITATIONS

The AII is not capable of creating SAS System syntax necessary for creating data sets where the observations in the raw data sources span more that one row. The macro variable names that resolve to the file path name (change section 1) must be of the form: two identifying characters (A-Z) with a concatenated number that indicates 1 to the number of data sources. More than 9 data sources will create an error. Also, the value of the record indicator in the raw data source must be a maximum of two characters (A-Z) and the sub-record indicator must be a number 0-9. There also must be a single record and sub record indicator variable in the data source of the format previously discussed. There are probably many limitations to the AII that have not been explored, since it was conceived and created for a specific context. Yet, the benefits for adopting a similar strategy using SAS macro in other contexts to achieve the same result seem promising.

CONCLUSION

The AII was conceived to capitalize on redundancy in SAS System syntax. The redundancy occurred because hundreds of programs were created over time for different purposes at a public agency, yet every project required the initial creation of a SAS System data set(s) from the same set of raw data sources. In a shared mainframe environment, the AII provided the most parsimonious solution. A relational database solution was not an option due to cost and lack of memory. Yet, continuing a short sighted project-by-project approach to data processing incurred a time penalty through consecutive rather than concurrent program execution, redundant creation of SAS System data sets across projects, and unnecessary debugging when redundant syntax was adapted for new projects. The lack of planning also resulted in little standardization of SAS System data sets across projects and led to an inability to capitalize or to even recognize redundancy in data re-coding, re-structuring, merging, routing, or reporting. The AII provides a tool for creating standardization in program development and a basis for expanded automated informational management in a non-relational database environment.

TRADEMARKS

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

S. Patrick Thornton
Child Services Research Group
44 Montgomery Street Suite 1450
San Francisco, CA 94104-4602
(415) 502-8004
SPThornton1@juno.com

FIGURE 1. AII: EXAMPLE CONFIGURATION

```
**************************************************************************;
*Automated Input Interface
 Created by S. Patrick Thornton
 University of California San Francisco;

*Note: This is a PC demonstration, however the AII was originally
conceived as a tool for programmers working with SAS in Batch
mode in an MVS environment;
**************************************************************************;
OPTIONS MPRINT;

*Includes the SAS Macros that creates the user specified data
sets and input statements;

%INCLUDE 'C:\Projects\SAS Presentation\AII
Complete\allds.sas';
%INCLUDE 'C:\Projects\SAS Presentation\AII
Complete\alltap.sas';
**************************************************************************;
*Change Section 1;
*Locations of raw data;
%LET CM1 = C:\PROJECTS\SAS PRESENTATION\CM1.TXT;
%LET CM2 = C:\PROJECTS\SAS PRESENTATION\CM2.TXT;
%LET PM1 = C:\PROJECTS\SAS PRESENTATION\PM1.TXT;
```

```
%LET PM2 = C:\PROJECTS\SAS PRESENTATION\PM2.TXT;

*************************************************************************;
*Change Section 2;
*AII Control Data Set;
*This is a demonstration Control Data Set, but in actual practice
this would be much longer and stored in a permanent SAS
Library;
DATA TEMP2;
  INPUT TAP $ RECOR $ SUBREC $ VARN $ VARL $ C  DAF $
reci $;
  IF DAF = 'H' THEN DAF = '';
  CARDS;
  CM AB 0 spn              23-30     1          H      H
  CM AB 0 CDI              2-4       1          H      P
  CM AB 0 CAS              5-16      1          H      P
  CM AB 0 OFF              $31-40    1          H      H
  CM AB 0 OFFDAT           @41       2 MMDDYY6.    H
  CM AB 0 RTY              $17-18    1          H      R
  CM AB 0 SNU              19-21     1          H      P
  CM AB 0 CCN              22        1          H      S
  PM AZ 1 SPN              1-10      1          H      P
  PM AZ 1 NAME             $11-20    1          H      H
  PM AZ 1 GENDER           21-22     1          H      H
  CM LQ 1 CDI              2-4       1          H      P
  CM LQ 1 CAS              5-16      1          H      P
  CM LQ 1 RTY              $17-18    1          H      R
  CM LQ 1 SNU              19-21     1          H      P
  CM LQ 1 CCN              22        1          H      S
  CM LQ 1 POI              $23-24    1          H      H
  CM LQ 1 POIDAT           @33       2 MMDDYY6.    H
  ;
*************************************************************************;
*Change Section 3;
*User Interface to allow selection of each data source where each
has the following components;
* physical location (t);
* number of physical locations (NT);
* number of records (NR) from the physical location;
* records and sub-records (R1..R10) of data at physical location;
DATA FIRST;
   INPUT T $ NT NR R1 $ R2 $ R3 $ R4 $ R5 $ R6 $ R7 $ R8 $
R9 $ R10 $;
   RETAIN K 0;
   K = K + 1;
   CALL SYMPUT('DS',K);
   DROP K;
   CARDS;
  CM 2 2 AB0 LQ1 H H H H H H H H
   ;

*************************************************************************;
*Change Section 4;
*User interface to allow the choice to select specific variables
from the records of each data source. Sel1-3 control the building
of input statements for the data source specified in the
corresponding observation of the data set above (first);

%GLOBAL SEL1;  *controls first data set in first;
 %GLOBAL SEL2; *controls second data set in first;
 %GLOBAL SEL3; *controls third data set in first;
*if you want to select set to 'yes';
*e.g. SEL1 controls first data set (obs) in first
(e.g. CM 2 2 AB0 LQ1);
 %LET SEL1 = ;
 %LET SEL2 = ;
 %LET SEL3 = ;

*************************************************************************;
*Change Section 5;
*Choose the specific variables from a data source, record, and
sub record to include in the input statments created by the
program;
*As specified in data set first set the following parameters:
*   the physical location (TAP);
*   the record (RECOR) from the physical location;
*   the sub record (SUBREC) from the physical location;
*   the variable literal names (V1..V15) and place holders (H);

DATA SEC;
  INPUT TAP $ RECOR $ SUBREC $ V1 $ V2 $ V3 $ V4 $ V5 $
V6 $ V7 $ V8 $ V9 $ V10 $ V11 $ V12 $ V13 $ V14 $ V15 $;
  CARDS;
  CM AB 0 SPN CDI OFF H H H H H H H H H H H
  CM LQ 1 CAS POI POIDAT H H H H H H H H H H H
  ;
*************************************************************************
;
%ALLDS(&DS, BOGUS);
```

FIGURE 2. AII: EXAMPLE LOG

```
MPRINT(ALLTAP):   DATA CMAB01 (KEEP
MPRINT(ALLTAP):  = SPN CDI CAS OFF OFFDAT RTY SNU
CCN ) CMLQ11 (KEEP
MPRINT(ALLTAP):  = CDI CAS RTY SNU CCN POI POIDAT );
MPRINT(ALLTAP):   INFILE "C:\PROJECTS\SAS
PRESENTATION\CM1.TXT";
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   CDI 2-4 CAS 5-16 RTY $17-18 SNU 19-21
CCN 22
MPRINT(ALLTAP):   @;
MPRINT(ALLTAP):   *IF _N_ > 40000 THEN STOP;
MPRINT(PUTALL):   IF RTY = "AB" THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(PUTALL):   IF CCN = 0 THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   SPN 23-30 OFF $31-40 @41 OFFDAT
MMDDYY6.
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   OUTPUT CMAB01 ;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   END;
MPRINT(PUTALL):   IF RTY = "LQ" THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(PUTALL):   IF CCN = 1 THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   POI $23-24 @33 POIDAT MMDDYY6.
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   OUTPUT CMLQ11 ;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   RUN;

NOTE: The infile "C:\PROJECTS\SAS
PRESENTATION\CM1.TXT" is:
      FILENAME=C:\PROJECTS\SAS PRESENTATION\CM1.TXT,
      RECFM=V,LRECL=256

NOTE: 10 records were read from the infile "C:\PROJECTS\SAS
PRESENTATION\CM1.TXT".
```

```
      The minimum record length was 0.
      The maximum record length was 47.
NOTE: SAS went to a new line when INPUT statement reached
past the end of a line.
NOTE: The data set WORK.CMAB01 has 9 observations and 8
variables.
NOTE: The data set WORK.CMLQ11 has 0 observations and 7
variables.
NOTE: The DATA statement used 0.38 seconds.

MPRINT(ALLTAP):   DATA CMAB02 (KEEP
MPRINT(ALLTAP):  = SPN CDI CAS OFF OFFDAT RTY SNU
CCN ) CMLQ12 (KEEP
MPRINT(ALLTAP):  = CDI CAS RTY SNU CCN POI POIDAT );
MPRINT(ALLTAP):   INFILE "C:\PROJECTS\SAS
PRESENTATION\CM2.TXT";
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   CDI 2-4 CAS 5-16 RTY $17-18 SNU 19-21
CCN 22
MPRINT(ALLTAP):   @;
MPRINT(ALLTAP):   *IF _N_ > 40000 THEN STOP;
MPRINT(PUTALL):   IF RTY = "AB" THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(PUTALL):   IF CCN = 0 THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   SPN 23-30 OFF $31-40 @41 OFFDAT
MMDDYY6.
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   OUTPUT CMAB02 ;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   END;
MPRINT(PUTALL):   IF RTY = "LQ" THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(PUTALL):   IF CCN = 1 THEN DO;
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   INPUT
MPRINT(PUTALL):   POI $23-24 @33 POIDAT MMDDYY6.
MPRINT(ALLTAP):  ;
MPRINT(ALLTAP):   OUTPUT CMLQ12 ;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   END;
MPRINT(ALLTAP):   RUN;

NOTE: The infile "C:\PROJECTS\SAS
PRESENTATION\CM2.TXT" is:
      FILENAME=C:\PROJECTS\SAS PRESENTATION\CM2.TXT,
      RECFM=V,LRECL=256

NOTE: 18 records were read from the infile "C:\PROJECTS\SAS
PRESENTATION\CM2.TXT".
      The minimum record length was 38.
      The maximum record length was 47.
NOTE: The data set WORK.CMAB02 has 10 observations and 8
variables.
NOTE: The data set WORK.CMLQ12 has 8 observations and 7
variables.
NOTE: The DATA statement used 0.39 seconds.

MPRINT(ALLTAP):   PROC APPEND BASE=ACMAB01
DATA=CMAB01;
MPRINT(ALLTAP):   RUN;

NOTE: Appending WORK.CMAB01 to WORK.ACMAB01.
NOTE: BASE data set does not exist. DATA file is being copied
to BASE file.
NOTE: The data set WORK.ACMAB01 has 9 observations and 8
variables.
NOTE: The PROCEDURE APPEND used 0.05 seconds.

MPRINT(ALLTAP):   PROC APPEND BASE=ACMAB01
DATA=CMAB02;
MPRINT(ALLTAP):   RUN;

NOTE: Appending WORK.CMAB02 to WORK.ACMAB01.
NOTE: 10 observations added.
NOTE: The data set WORK.ACMAB01 has 19 observations and
8 variables.
NOTE: The PROCEDURE APPEND used 0.05 seconds.

MPRINT(ALLTAP):   PROC PRINT DATA=ACMAB01;
MPRINT(ALLTAP):   TITLE "aCMAB01";
MPRINT(ALLTAP):   RUN;

NOTE: The PROCEDURE PRINT used 0.0 seconds.

MPRINT(ALLTAP):   PROC APPEND BASE=ACMLQ11
DATA=CMLQ11;
MPRINT(ALLTAP):   RUN;

NOTE: Appending WORK.CMLQ11 to WORK.ACMLQ11.
NOTE: BASE data set does not exist. DATA file is being copied
to BASE file.
NOTE: The data set WORK.ACMLQ11 has 0 observations and 7
variables.
NOTE: The PROCEDURE APPEND used 0.05 seconds.

MPRINT(ALLTAP):   PROC APPEND BASE=ACMLQ11
DATA=CMLQ12;
MPRINT(ALLTAP):   RUN;

NOTE: Appending WORK.CMLQ12 to WORK.ACMLQ11.
NOTE: 8 observations added.
NOTE: The data set WORK.ACMLQ11 has 8 observations and 7
variables.
NOTE: The PROCEDURE APPEND used 0.05 seconds.

MPRINT(ALLTAP):   PROC PRINT DATA=ACMLQ11;
MPRINT(ALLTAP):   TITLE "aCMLQ11";
MPRINT(ALLTAP):   RUN;
NOTE: The PROCEDURE PRINT used 0.0 seconds.
```