Creating a Shipping Information System Using SAS® Software and Bar Codes
Sarah Howey Dietrich, SAS Institute Inc., Cary, NC

ABSTRACT
A unique shipping information system was designed using SAS® software and bar coded labels to get quick and accurate information. Bar coding data saved time, reduced errors, and eliminated keying. SAS software was used to read and process the bar coded information, producing reports for UPS® (United Parcel Service), RPS® (Roadway Package System), other shippers, and SAS institute Inc. management. A master shipping data base was created to track packages. The system allows the user to have immediate enhancements as the postal regulations and rates change and provides the best service at the least cost for the institute and customer.

INTRODUCTION
This paper describes the development of an in-house shipping information system that maintains a logistical record of all outgoing packages from the Institute. The challenge was to develop a shipping application that
- eliminated hand entering data
- minimized errors
- minimized downtime
- required minimal additional programming
- required minimal maintenance
- allowed postal rates to be updated with minimal effort
- captured data from all outgoing packages for managerial reports, for tracing lost packages, and for statistical analysis to ensure that the company and customers are getting the best deal for the money.

When orders are placed, the necessary order information including the shipping address and method is entered in a SAS data set using PROC FSEDIT. These orders are processed to generate invoices for the customer. At the time invoices are generated, shipping labels are produced with the pertinent information bar coded. The bar codes are used in our in-house shipping system to eliminate rekeying data, to reduce errors, and to save time. The shipping labels and the invoices are printed in the same order so that little time is required to match them. A copy of the invoice serves as the packing slip. Once the package is packed and the shipping label is affixed, the package is then ready to be entered into the shipping information system. The package moves down a conveyor belt where it is placed on a weighing scale. The bar code on the label is read and transferred to the FSEDIT screen. The bar code reader automatically reads the weight from the bar code printer and paper selection
The first step was to evaluate the quality and quantity of bar codes produced by the printers currently used by the Institute. Finding quality paper with a sticky backing that could be used for the bar codes required much testing. The bar code paper had to meet standards for not smearing, durability, reading legibility, and printing compatibility with a laser printer. A laser printer was selected that
- used different paper sizes and types (transparencies, labels)
- printed quietly at a speed of ten pages per minute
- combined high quality resolution character formation with a wide variety of type styles (referred to as fonts).

The decision was made to purchase the bar code font in cartridge form so the font would always be in the memory of the printer and would not have to be downloaded each time the printer was powered off.

HARDWARE SELECTION
The next step was to select compatible hardware including a weighing scale, a bar code reader, a terminal, and a continuous form printer. In the search, scales and bar code readers that interfaced with terminals currently in use at the Institute were given higher consideration. The bar code reader eliminates keying, saves time, and reduces errors by electronically reading data that would normally be entered through a keyboard. A bar code reader was needed that provided the flexibility of reading a variety of bar codes. The shipping information system would use two types of bar codes, 3 of 9 bar codes generated by the Institute as well as the RPS interleave 2 of 5 bar codes. The goal was to select the hardware that best met our requirements at the best price.

The hardware selection process was time consuming and frustrating. Many vendors sell bar code readers, but few will interact with a terminal and a scale. Even fewer bar code readers worked in a test application. Finally, a company was located that produced a bar code reader that would interact with the terminal already in use and with the model scale selected to meet postal regulations. A printer was purchased to print the manifest listing, which must be printed on continuous-form paper and must be given to the shipper along with the packages. The next step was programming the shipping application using SAS software.

PROGRAMMING THE SHIPPING INFORMATION SYSTEM
The programming process was accelerated by knowing the capabilities of the SAS System and the type of application needed by those processing the outgoing packages. The shipping information system screen was developed by using SAS macro facilities.
within SAS/AF® software. Data are entered using the FSEDIT procedure in SAS/FSP® software. User-written informats provide instantaneous error checks for inputted data, and user-written formats are used to create new data values from existing data. When developing the shipping information system, main concerns of the application developer included

- error checking
- speed and efficiency
- ease of use
- low maintenance.

Replacing IF-THEN statements with user-written formats and informats is the secret to the speed, efficiency, and minimal errors in the shipping information system. See Chapters 15 and 36 in the SAS User’s Guide: Basics, Version 5 Edition for a description of the procedures to use for standard SAS formats and informats as well as how to generate user-written formats. In Release 5.18 of the SAS System, you can create your own informats with the INVALUE statement in the FORMAT procedure. Informats are created in a similar way to formats with some special key words to help in error checking. Refer to SAS Technical Report P-175, Changes and Enhancements to the SAS System, Release 5.18, for more information about user-written informats.

Formats and Informats

User-written informats convert input values into a different form; user-written formats display data values in a different form. Both formats and informats can function as a table lookup for data validation. For example, you can

- validate data
- convert a digit to a character string
- convert a character string to a number
- convert a character string to a different character string.

What are User-written Informats? An informat is a set of directions for reading a value. The informats can convert character values into real numeric values, but the informats cannot convert real numbers into characters. Both standard SAS informats and user-written informats can be associated with variables in the

- DATA step using INPUT, ATTRIB, or INFORMAT statements and the INPUT function
- PROC step using an INFORMAT statement.

A typical use of the user-written informats in the shipping information system is using the INFORMAT statement with the FSEDIT procedure where new data values are entered. Data values are entered into a SAS data set accessed as follows:

```
PROC FSEDIT DATA=X.X SCREEN=X.XCH;
INFORMAT SHIPHOW $SHIPHOW.
DATA Ai;
```

When the data are entered into the variable SHIPHOW, the value is verified by the $SHIPHOW informat. The user-written informat called $SHIPHOW created by the FORMAT procedure is

```
PROC FORMAT;
INVALUE $SHIPHOW
"* SHippers specific to our company */
"PAR", /* Shipper optional - set by program */
"UPS", /* United Parcel Service */
"EPD" = $SAME; /* E.P.D. = $SAME */
OTHER = _ERROR; /* Data value is not valid */
```

The _ERROR_ specification is used in combination with _SAME_ for performing data validation. In a character informat, _SAME_ causes the SAS System to store the values in the range without change. If the informat is numeric, the SAS System tries to read the values as numbers; if that is not possible, the SAS System assigns a missing value and issues an invalid data message. _SAME_ is the default action when no informatted value is given. _SAME_ is used for indicating several informatted values without listing each of them. For example, the GROUP informat reads values of 01 through 20 and assigns the numbers 1 through 20 as the result:

```
PROC FORMAT;
INVALUE GROUP
"01" - "26" = $SAME;
```

If a value is entered for a variable with an informat attached and a match is not found in the informat, then the _ERROR_ causes the SAS System to treat these values in the designated range as invalid data. The SAS System returns the message

ERROR: data value is not valid. Please reenter.

for invalid data entered in the variable on the FSEDIT screen. The observation will be held until an acceptable value is entered for the informatted variable or a cancel command is entered on the command line to delete the record. Multiple variables can be concatenated to create informats and to enter data for error checking. These concatenated multiple variables allow for multiple data validation. For example, the state and ZIP code values are combined and entered into the variable STZIP so the user-written informat called $OSSTZIP can verify that the state and ZIP code ranges are correct. State and ZIP code discrepancy checks for the correct ZIP code range with the correct state are instantaneous. Without the informat capabilities, processing would have to be done in a DATA step, and data validation is not instantaneous. The code to create the informat $OSSTZIP is

```
PROC FORMAT;
INVALUE $OSSTZIP
"*/ ZIP code ranges for each state in USA */
"US35000" - "US40999", /* Alabama ZIP code range */
"FL35000" - "FL40999", /* Florida ZIP code range */
"WI35000" - "WI40999" = _SAME_, /* Wisconsin ZIP code range */
OTHER = _ERROR; /* Data Value is not valid */
```

What are User-written Formats? A format is a set of directions for writing or printing a value. Both standard SAS formats and user-written formats can be associated with variables in the

- DATA step using PUT or FORMAT statements
- PROC step using a FORMAT statement.

A typical use of the user-written formats in the shipping information system is with the PUT function in a DATA step where new data values need to be created from the original input values. As shown in the DATA step below, the PUT function allows you to create a value for the variable ZONE based on the match in $UPS format for the first three characters of ZIPCODE variable:

```
PROC FORMAT;
VALUE UPS
000 - 065 = 4
066 - 299 = 4
315 - 339 = 4
340 - 354 = 5
355 - 999 = 4
OTHER = _ERROR;
DATA Ai;
ZONE = /* DATA step to create zone */
SET PSHPNLOADER;
IF SHIPHOW='UPS' THEN ZONE=PUT(SUBSTR(ZIP code,1,3),$UPS.);
```

478
ELSE IF SHIPHOW='RPS' THEN
  ZONE=Ptrr{SUBSTRjZIP code,l,3},$RPS.J;
RUN;

The variable ZONE corresponds to a zone number assigned to a geographical area by the federal postal system based on the distance from the package origination point. Each shipper provides a chart of the ZIP codes within the zone where they deliver packages. ZONE is set to missing if no match is found in the ZONE formats. An error check brings the observation to your attention in the shipping information system. The zone and weight of each package are necessary to determine the base postal charge.

Postal Rate Changes

To update the postal changes with minimal effort, a SAS data set similar to the one shown in Screen 1 was created. The data set contains the zone, weight, and postal rate for packages weighing from 1 to 70 pounds. The end user changes the SAS data set in FSEDIT.

Notice again the combination of multiple values to create the $SHPACT format. WEIGHT is concatenated with ZONE on the left-hand side and set equal to the postal charge on the right-hand side. BASECHG is created by using the PUT function with the $SHPACT format attached to the variable, which is weight concatenated with zone. If no match occurs, then BASECHG is set to zero. An error routine checks for this exception. The shipping application will continue to check for errors until all errors have been corrected. A minimum amount of code is needed to do the processing when user-written formats are used. For example, if ZIP code is equal to 32801 and WEIGHT is equal to one pound, the following code using the PUT function along with standard and user-written formats will set: ZONE=4 and BASECHG=1.60.

Example: ZIP code = 32801 WEIGHT = 1 pound
DATA A;
  SET PSHPSH1.0RDER;
  /* ZIP code to zone SAS code using formats */
  BASECHG = PUT(WEIGHT,Z2.jI lZONE),$SHPACT')i; 
RUN;

Screen 1 Postal Rate SAS Data Set

A SAS program similar to the one below is used to create the format each time the previous data set is updated.

```
DATA ...NULL;
  SET SHIPRATE.RATES END="EOF;" /* data set maintained by user */
  FILE FORMAT1;
RUN;

WTZONE = PUT(HEIGHT,Z2. )I I ZONE;
IF ...!L=1 THEN DO;
  PUT ' PROC FORMAT LIBRARY=LIBRARY;';
  PUT ' VALUE SSHPACT';
END;
PUT 'WTZONE'= 'BASECHG';
IF EOF THEN DO;
  PUT ' OTHER = ' '0.00';
END;
UNINCLUDE FORMAT1; /* Run in AF system by user when changes made */
```

Thus, the user can enter postal changes and create formats as well as informats while the developer continues to work on other projects. Almost no maintenance is needed by the programming staff. The format created interactively by the SAS program above is as follows:

```
PROC FORMAT LIBRARY=LIBRARY;
  VALUE $SHPACT
  /* weight concatenated with zone equals basechq */
  '01A' = '13.00 ';
  '01B' = '15.61 ';
  '01C' = '15.89 ';
  '01D' = '16.53 ';
  '01E' = '17.95 ';
  '01F' = '11.50 ';
  '016' = '11.50 ';
  '01G' = ' 8.50 ';
  '01H' = ' 9.50 ';
  '01J' = '12.00 ';
  '01K' = '12.00 ';
  '01L' = '13.00 ';
  '01M' = '13.00 ';
  '01N' = '14.50 ';
  '01O' = ' 8.00 ';
  '01P' = ' 8.00 ';
  '01Q' = ' 8.00 ';
  '01R' = '22.00 ';
  '01S' = ' 6.50 ';
  '01T' = ' 5.50 ';
  '01U' = ' 4.00 ';
  '01V' = ' 4.00 ';
  '01W' = ' 2.00 ';
  '01X' = ' 1.75 ';
  '01Y' = ' 1.75 ';
  '01Z' = ' 1.75 ';
  '011' = ' 1.75 ';
  '012' = ' 1.75 ';
  '013' = ' 1.75 ';
  '014' = ' 1.75 ';
  '015' = ' 1.75 ';
  '016' = ' 1.75 ';
  '017' = ' 1.75 ';
  '018' = ' 1.75 ';
  '019' = ' 1.75 ';
  '01A' = ' 8.50 ';
  '01B' = ' 8.50 ';
  '01C' = ' 8.50 ';
  '01D' = ' 8.50 ';
  '01E' = ' 8.50 ';
  '01F' = ' 8.50 ';
  '01G' = ' 8.50 ';
  '01H' = ' 8.50 ';
  '01I' = ' 8.50 ';
  '01J' = ' 8.50 ';
  '01K' = ' 8.50 ';
  '01L' = ' 8.50 ';
  '01M' = ' 8.50 ';
  '01N' = ' 8.50 ';
  '01O' = ' 8.50 ';
OTHER = ' 0.00 ';
```

Notice again the combination of multiple values to create the $SHPACT format. WEIGHT is concatenated with ZONE on the left-hand side and set equal to the postal charge on the right-hand side. BASECHG is created by using the PUT function with the $SHPACT format attached to the variable, which is weight concatenated with zone. If no match occurs, then BASECHG is set to zero. An error routine checks for this exception. The shipping application will continue to check for errors until all errors have been corrected. A minimum amount of code is needed to do the processing when user-written formats are used. For example, if ZIP code is equal to 32801 and WEIGHT is equal to one pound, the following code using the PUT function along with standard and user-written formats will set: ZONE=4 and BASECHG=1.60.

Example: ZIP code = 32801 WEIGHT = 1 pound
DATA A;
  SET PSHPSH1.0RDER;
  /* ZIP code to zone SAS code using formats */
  BASECHG = PUT(WEIGHT,Z2.jI lZONE),$SHPACT; );
RUN;

Layout of Package Label

Before this system was implemented, it took three labels attached to each package to give the information now contained on a single bar code label. While cost increased by half a cent per new label, the error margin and time spent matching three labels per package quickly negated the increased cost. Currently, the bar code label generated using SAS software for each package reads

```
SAS Institute Inc.
SAS Campus Pkwy.
Cary, NC 27513-8000
Phone (919) 585-9900
Fax (919) 585-1609

* UPS SHIPPER NO. *
* MC 204-454 *
* REG 10F 274560 *

*
*
*
*
*
*
*

* UPS 274560
dsupplier 512-758-5171
SAS INSTITUTE INC
ATTN: RON MILLER
12873 RESEARCH BLVD.
AUSTIN, TX 78759

* 21058766 62/14640 95 *

Figure 1 Shipping Label
```
Shippers require different information on the shipping label. The company shipper number and the unique package identification number is enclosed in a box in the upper-right corner of this label. The address information appears in the mid-section of the label. The Institute uses the first line of the address to designate what department sent the package. In the event the package is returned, it can easily be forwarded to the correct sender. The shipping method, ZIP code, zone, unique package identification number, and department are coded in bar codes. The bar code contains all the information that is needed for the in-house shipping information system.

**How to Create Bar Code Labels**

The layout of the label is done using PUT and FILE statements in a DATA step. The N=PS option is specified in the FILE statement. With the N=PS option, all lines on the page are available to the pointer so each page of output can be arranged before writing it. The row and column variables defined in the SAS program mark the current line and column of the pointer. The SET statement reads a SAS data set containing the values. When the last value in the last column has been placed, the PUT statement writes the entire page. The labels are printed by placing within the SAS program the code that tells the laser printer which font to use to print the bar code and which font to use to print the other information. SAS software easily communicates font changes.

**Ease to Add or Delete Shipper**

Since shippers and methods are always changing, the shipping information system must be flexible for adding or deleting shippers. Shippers can be easily added or deleted by updating the shipper informat, ZIP code to zone formats related to shipper, and base postal charge format. Additional programming is minimal for the programming staff.

**Manifest Designed to Shipper Specifications**

The manifest is a listing for each shipper of all the packages shipped on a certain day. In order to comply with the shipper’s regulations, the packages must be listed on the manifest in the order they were entered into the shipping information system. This manifest must be presented at the time the packages are picked up by the shipper. Thus, for the address information to be present on the manifest, the data set created by reading the bar codes on the packages must be merged by the unique package identification number with an address data set. Bar coding the address information was not practical. Too much space was required on the label, and the data could be merged faster than it would be read by the bar code reader.

**FINANCIAL BENEFITS**

Cash outlay for implementing this project was as follows:

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>$1,505.00</td>
</tr>
<tr>
<td>Bar code reader</td>
<td>760.00</td>
</tr>
<tr>
<td>Continuous-form printer</td>
<td>1,795.00</td>
</tr>
<tr>
<td>Bar code font cartridge</td>
<td>200.00</td>
</tr>
<tr>
<td>Laser printer</td>
<td>4,000.00</td>
</tr>
<tr>
<td>Subtotal (one time cost)</td>
<td>$8,350.00</td>
</tr>
<tr>
<td>Less: Yearly leased system cost</td>
<td>$14,195.52</td>
</tr>
<tr>
<td>Savings to the Institute in first year</td>
<td>$5,845.52</td>
</tr>
</tbody>
</table>

The laser printer had been purchased for use with another project. Currently, the laser printer capabilities are being fully utilized with the addition of bar code printing. The in-house shipping information system allows the Institute to receive competitive bids for shipper service. The Institute will continue to save approximately $14,200 per year due to the development of the shipping information system and will continue to have:

- timely updates for postal changes
- accurate financial reports
- actual data for statistical analysis and comparisons
- minimal downtime
- minimal data entry and errors.

**CONCLUSION**

The key to the success of the shipping information system is the use of user-written formats and informats that return table values instantaneously. The shipping information system has been a wise investment for the Institute: employee morale is high with the new shipping information system; the Institute has broken even in less than a year; and most importantly, the company has a system that can be quickly updated for changes to meet processing needs in a timely and efficient manner. The shipping system has been so successful that a company-wide system was developed using SAS/AF software. Now all packages, whether business or personal, can be sent using the shipping information system.

**REFERENCES**


SAS, SAS/AF, and SAS/FSP are registered trademarks of SAS Institute Inc., Cary, NC, USA.

UPS is a registered trademark of United Parcel Service, Greenwich, CT, USA.

RPS is a registered trademark of Roadway Package System, Pittsburgh, PA, USA.