FUEL FOR THOUGHT

Application Development Using SAS® software: A case study

Bruce Leister,
System Development Officer,
Corporate Transport Department,
State Electricity Commission of Victoria.

Background

The State Electricity Commission of Victoria operates a large IBM compatible mainframe, with TSO as the main operation system for small applications. There are approximately 1,000 TSO users.

SAS software has been used within the SECV since 1981 for enduser computing, programming, and capacity planning.

A number of small systems have been developed over the years - either as prototypes, or as final systems, utilising SAS software as the main data processing language.

WRS Work Reporting System

A project tracking task allocation and timesheet recording system, used internally to Information Systems Dept. (ISD).

EO Information Systems Department’s Cost Reporting System

Collects information on services provided to client departments, produces reports and transfers the data to the general ledger for charging.

IPO Interim Purchase Order System

A system to automate and speed the generation of purchase orders throughout the SECV. SAS software is used in the report/order generation only.

EU Fuel Credit Card System

A system to read an invoice tape, verify internal correctness, and produce transactions for the Material Management System (MMS).

The FUEL system (SHELL card)

In December 1987, the SECV started using the SHELL CARD credit card for all fuel purchases from resellers.

The SECV currently spends between $250,000 and $400,000 per month on credit card purchases from external resellers. Thus the need for a system for accurate and timely processing of monthly invoices.

In November 1987, I started the analysis phase of the development of a computer system to process a monthly tape of transactions:

- read and verify the tape for consistency
- verify registration numbers with our inventory
- verify pricing details
- generate an invoice
- generate transaction file for the Material Management System

The main menu screen looks like that shown in figure 1. The tools used to achieve the quick development time were:

SAS software: For all data processing & the major reporting functions

SAS software was used because of the ease of coding complex data manipulation tasks and reading/writing foreign files.

ISPF: For the menu system & user interface, tailoring & submission of batch jobs

ISPF was used because of the ease of screen design, generation and user friendly interface, as well as its ‘quick’ user response.

PL/I & CLIST: The driving programs behind ISPF

PL/I was used for its speed of execution.

All the above tools allow easy cloning of code, thus reducing development time.

Prototyping is also easy, as the screens can be designed, coded and confirmed while the processing code is still on the drawing board.

The system is structured around ISPF menus & functions which either submit a batch SAS job to perform processing or display a report dataset or ISPF table (output from a previous SAS job).

The SECV uses a data dictionary of all data elements, screens, and files, thus all stored variables are of the form D15205, etc. which is meaningless without the corresponding dictionary. Some of the variables used are listed below:

- D15205 - product code
- D15206 - transaction unit price
- D15207 - fuel quantity
- D15208 - transaction price
- D15209 - reseller’s name
- D15212 - registration number
- D15226 - our base price
- D15234 - MMS batch number

Read and verify the tape

This step of the process reads a record from the tape, depending on the record type performs internal validation and writes the record out.

Checks are made for duplicate transactions, these are held in a suspense table awaiting operator intervention.

Verify registration numbers with our inventory

A extract is made of our inventory database (a FOCUS database) which is then compared for unknown registration numbers, these are held in another suspense table.

Verify pricing details

Our contract price is made up of two components, a base price and a freight differential, based on distance from the nearest depot.

The transactions have the name of the reseller, a table linking reseller names to town names, and a town to freight price is maintained. See Figures 2 & 3.

When a town name is entered for a new reseller location, a cross-reference check is made to ensure the town is a known one. If not then a new town has to be entered and the freight differentials entered.

The tables (figures 2 & 3) are used to calculate the expected price, this is then compared to the actual, any transaction where they differ is reported awaiting operator action. A check for unknown towns has to be made, else freight differentials can not be calculated.
* EU300 - verify prices & locations;*

```sas
*------------------------------------------------;
* data f02590; * read in location / town table;
  infile f02590;
  attrib d15222 length=24; * reseller location
  attrib d15209 length=22; * name of the town
  if _n_ = 1 then do;
    d15222 = 'ADMIN CHARGE';
    d15209 = d15222; output;
  end;
  input d15222 $24~ @25 d15209 $22.;
  if d15222 = ' ' and d15209 = ' ' then delete;
  output;
*------------------------------------------------;
```

```sas
proc sort;
  by d15209;
*------------------------------------------------;
```

```sas
data unknown (keep=d15209)
  step1 (rename={d15203-d15233});  * locations ?
  merge f02584., supplier {in=a f02590 {in-b};
  by d15209;
  if a; then do;
    if first.d15209 then output unknown;
    end;
  else output step1;
  proc sort data=unknown nodup;
  by d15209;
```

```sas
Generate an invoice and transaction file

Once the transactions have passed the price test, an invoice is printed, so Shell can be paid, and the necessary transactions are formatted for entry into the MMS and thought into the cost accounting system.

See below for an example of the code used to generate the transactions.

**Formats and the PUT function**

Use is made of SAS formats and the PUT function in converting coded values to values to be used. One example of this technique is the Product table, by translating this table into a series of SAS formats, it is possible to use the product code to determine, the price/fee, the Stock Reference Number, or its description.

The Product table is maintained in an ISPF table and dumped into a SAS dataset as well as create some formats whenever it is altered, see figure 3.

**Code to Build a Table**

```sas
* EU400 - create product database in SAS ;
* and a series of formats ;
* options missing = ' ';
proc pds ddbname=library;
  * delete old formats ;
    delete eurname eorder eitem eustock;
```

```sas
data temp; * read the new product output file;
  infile f02591;
  attrib d15233 format=ddmmyy8.;
  label = 'Date of price change';
  input D15205 d15226 D15220 D15221 D15220 D15221 D15220 D15233 ddmmyy8.
  @10 D15239 $11.
  @10 D15233 ddmmyy8.
  @10 D15239 $11.
  proc sort;
  by D15205 D15233;
  proc sort data=f02585.product out=temp2;
  by D15205 D15233;
```

```sas
Using the Format that has been created

```sas
* EU400 - create MMS transactions and invoices ;
* data stock nonstock;
* split into stock and nonstock items ;
* this is for charging purposes;
  set f02585.supplier;
  d15207 = round(d15207,1);
  if put(d15207,eustock.) = ' ' then output nonstock;
  else output stock;
  %include 'eucbatch';
  * get batch control numbers and put into MACROS;
proc summary data-issue;
  * create summary for batch control ;
class d15212 d15205;
  var d15207;
  output out-issue sum-d15207;
```
The solution (or two)

SAS/DIMI

This product is SAS/Dialog Manager Interface, which is another name for ISPF, allows a SAS data step to call ISPF functions and read/write ISPF tables directly.

It even allows SAS software to be the driving language behind the user-interface. Though with some lose in response.

The SFCV has plans to evaluate this product in the near future.

SAS/AF

Version 5 of SAS/AF does not have the functionality to be considered.

Version 6, with its many new & enhanced features, can provide all of the features (or comparable ones) to ISPF.

The whole system could be written in SAS now! The generation and tailoring of batch jobs could be messy, but workable.

Conclusion

Currently SFCV has 4 significant small systems in operation, thus proving SAS software is an extremely useful tool in small system application development. The addition of SAS/DIMI and the new features of version 6 can only improve the SAS software product as a favored development tool for small applications.

SAS, SAS/AF, SAS/DIMI are registered trademarks of SAS Institute Inc., Cary, NC, USA.
### Freight Differentials

<table>
<thead>
<tr>
<th>Sel</th>
<th>Name of town</th>
<th>LPG</th>
<th>Motor</th>
<th>Distillate</th>
<th>Date of effect</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ADELAIDE</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>AIREY'S INLET</td>
<td>1.10</td>
<td>1.30</td>
<td>1.30</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALBERTON</td>
<td>2.70</td>
<td>3.00</td>
<td>3.00</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALBURY/WODONGA</td>
<td>3.20</td>
<td>1.60</td>
<td>1.80</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALEXANDRA</td>
<td>1.70</td>
<td>2.40</td>
<td>2.70</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALLANSFORD</td>
<td>2.50</td>
<td>2.00</td>
<td>2.30</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALLESTREE</td>
<td>0.70</td>
<td>1.00</td>
<td>1.00</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ALTONVILLE</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ANGLESEA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>APOLLO BAY</td>
<td>2.20</td>
<td>2.00</td>
<td>2.20</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>APSLEY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ARARAT</td>
<td>2.50</td>
<td>1.50</td>
<td>1.70</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>ARATULA</td>
<td>2.60</td>
<td>2.30</td>
<td>2.30</td>
<td>01/12/87</td>
</tr>
<tr>
<td></td>
<td>AYRNSDALE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AVENEL</td>
<td>1.60</td>
<td>1.80</td>
<td>1.80</td>
<td>01/12/87</td>
</tr>
</tbody>
</table>

Figure 2.

### Locations Database

<table>
<thead>
<tr>
<th>Sel</th>
<th>Purchase location</th>
<th>Name of town</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AVONDALE HEIGHTS A/P</td>
<td>MELBOURNE</td>
</tr>
<tr>
<td></td>
<td>B &amp; R TYRE SERVICE</td>
<td>DAYLESFORD</td>
</tr>
<tr>
<td></td>
<td>BRAINTNSDALE</td>
<td>BRAINTNSDALE</td>
</tr>
<tr>
<td></td>
<td>BRAINTNSDALE A/B</td>
<td>BRAINTNSDALE</td>
</tr>
<tr>
<td></td>
<td>BALLINGER'S SERVICE STN</td>
<td>GURINDAR</td>
</tr>
<tr>
<td></td>
<td>BARRY'S SERVICE STN.</td>
<td>TENTERFIELD</td>
</tr>
<tr>
<td></td>
<td>BATERNAY SELF-SERVE</td>
<td>BATERNAY</td>
</tr>
<tr>
<td></td>
<td>BAY GARAGE</td>
<td>MELBOURNE</td>
</tr>
<tr>
<td></td>
<td>BENDIGO TRUCK STOP</td>
<td>BENDIGO</td>
</tr>
<tr>
<td></td>
<td>BENZTOOL AUTO SALES SV</td>
<td>MILDURA</td>
</tr>
<tr>
<td></td>
<td>BLACK MOUNTAIN R/HOUSE</td>
<td>BLACK MOUNTAIN</td>
</tr>
<tr>
<td></td>
<td>BLACKBURN SELF SERVE</td>
<td>MELBOURNE</td>
</tr>
<tr>
<td></td>
<td>BLUE GUM S/STN</td>
<td>MELBOURNE</td>
</tr>
<tr>
<td></td>
<td>BLUE GUM SERVICE STN</td>
<td>MELBOURNE</td>
</tr>
<tr>
<td></td>
<td>BOOGABILLA</td>
<td>BOOGABILLA</td>
</tr>
</tbody>
</table>

Figure 3.