

Dynamic, Data-Driven, Web-Based Applications To Satisfy An Ever Changing World

**Bernard R. Poisson, Systems Analyst
Master Sergeant, U.S. Air Force, San Antonio, TX**

ABSTRACT

Today, business and government organizations thrive on information. They are totally dependent on their continued ability to rapidly and efficiently access information in order to function. Remove or reduce the capacity to access needed information and the organization slows to an ebb or worse, stops dead in its tracks. The changing nature of the information required, and the need to leverage costs while at the same time increasing the availability of information, makes the design and coding of applications a progressively more daunting task. Using a few powerful SAS® functions and a modest change in design approach, you can create applications capable of satisfying the continually changing information needs of your organization, decrease costs, and increase information availability. Two equally important derived benefits that we can not overlook are those of increased portability of the application, and quick recovery after system failures.

INTRODUCTION

The Air Force Surveys Branch is required to perform between 12 and 15 telephone-based surveys called "Computer Assisted Telephone Interviews" or CATIs per year. The average number of personnel that must be successfully contacted and interviewed for a survey project of this nature to be statistically representative of the Air Force (AF) as a whole is approximately 400. Take into consideration these projects arrive only two to three weeks before the results are required in a formal report to the Air Staff or Air Force Chief of Staff (CSAF). One quickly realizes the need to conduct the data collection phase with utmost efficiency. This paper describes the previous manual and inefficient method contrasted by the new automated and efficient approaches to call tracking. It then demonstrates how to achieve the same degree of flexibility in any application.

THE TASKING

Typically, the branch chief receives a no-notice tasking via telephone from the Air Staff on a much needed survey project for the CSAF and there's not much time to get the answers. The CSAF wants to know how a change in policy is affecting the Air Force and he wants the answer yesterday so he's prepared when he testifies before Congress in two weeks.

WHOM DO WE CALL?

Our three survey analysts quickly begin resolving issues such as whether the chief is interested in a specific population such as officers, enlisted, military only, civilians, a combination thereof, or the entire force. An additional consideration is whether certain theatres of operation such as continental U.S., Pacific, Asia, and Europe are to be the target, a combination, or all theatres. The list of demographic information is paired with data elements contained within the AF master personnel system. Once all matches are obtained they are included in SAS® code that generates the sample in a SAS® dataset when executed.

THE TWO METHODS

What occurs with the sample dataset from this point is the principal topic of this discussion. We will first briefly describe the manual method of call tracking and then move on to the new SAS® application called 'CATI Survey Manager'.

THE MANUAL METHOD

In the past, the sample dataset was sorted predominantly on time zone. The reason for this is self-evident. Once sorted, the dataset was 'dumped' to paper resulting in pages and pages of output. These paper listings were broken out and then kept separated to help in the scheduling of outgoing calls to AF members.

Day-1: Calls are accomplished using a combination of AF members from the AF Personnel Center and contracted 'temp' employees. Status Information must be documented on each call such as bad numbers, members deployed (TDY), members off, members out of the office, members that were left a message to 'call back', etc. There was no 'standard' method of tracking this information. As each caller progressed through their respective lists, they devised their own methodology (what worked best for them) for keeping track of what was going on within their listings. The Survey Branch had no control of the methods used. As time progressed during the day, additional listings corresponding to appropriate time zones beginning their duty-day, were divvied out to the callers. This worked OK for the first day.

Day-2: Consider what happens when 120-150 calls go out on day-1. Eighty of the originating calls were successful, great! But a fraction hit bad numbers or personnel deployed. The balance of approximately 50-60 personnel in varying time zones were left messages requesting they call back a specific caller when they returned. Now factor in that half or more of the AF personnel helping on the first day couldn't help on the second. To make matters worse, two of the 'temps' call in sick or are otherwise unavailable and may or may not be replaced. If the 'temps' are replaced, their replacements have different names than previous individuals. Listings from varying regions are strewn on everyone's desks. There's no standard method of indicating call status. Calls are coming in for people that aren't around, and listings containing the inbound caller's information have been completely misplaced. Some of the previous day's callers are in for day-2, but they're on the telephone when some of their return calls come in, and they've resumed working the same lists. Now two callers need the same listing to obtain respondent's information. Pretty gory picture!

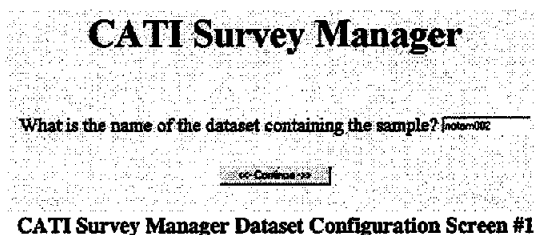
Days-3 & 4: A repeat of day-2 but slightly less confusing, but everyone is exhausted and not really looking forward to another confusing day. As the successful contacts add up to the required amounts, the data collection phase is ended.

The 'CATI SURVEY MANAGER' METHOD

Required Setup Procedures: Since a data-driven application will be used to perform tracking, there are a few simple configuration items that must be performed by the analyst before the initial call can be placed. They are:

1. The analyst calls up their web-browser on their desktop system and clicks the shortcut link to 'CATI Survey Manager'. The actual call to the broker invoked by the shortcut is:
`'http://srvyoni/vbin/surveybroker.exe?_program=catiprg.scl'`

Notice that no arguments are passed to the SAS[®] broker except for the '_program' argument, the minimum necessary for the application server to invoke the appropriate program. The analyst is greeted with the following screen:



No reference has been made to any mandatory dataset specifications except for the 5 'expected' data items included by default. **This is because the application can literally handle ANY SAS[®] dataset!**

2. After entering the name of the sample dataset, the analyst clicks on the 'Continue' button. This transmits the necessary information to the application via the broker. Now, the following screen is displayed:

CATI Survey Manager

(New Dataset Setup - Phase 1, Dataset NOTAM002)

The following filters are pre-defined within this application when the data items are contained within the survey sample dataset:

Time-Zone, SSN, Name, Rank and Gender

Additional data-items in the dataset NOTAM002 whose values can be used as filters are listed in the table below.

Select the the items below by which you wish to be able to filter:

Item to enable filtering on:		Filter
Dataset Variable Name	Dataset Variable Label	
RACE	RACE	☒
SOC	SOC	☒
OADD	ORGANIZATION	☒

No Additional Filters

Generate Filters!

CATI Survey Manager Dataset Configuration Screen #2

This dynamically generated HTML screen provides the analyst everything they need to know about the sample in an intuitive fashion. They are given the opportunity to specify additional items by name/label contained within the sample dataset to be used for filtering of caller data. Checking the corresponding check-box next to the data-item's name/label pair includes the item in the filter list. The following screen is generated when the analyst clicks 'Generate Filters', showing completion of the necessary configuration of the application for this CATI.

CATI Survey Manager

(New Dataset Setup - Phase 2, Dataset NOTAM002)

Creating (3) selected filters for dataset NOTAM002...

	Filter Name	Filter Selection
1	Race	4
2	SOC	2
3	Organization	468

<< Continue >>

CATI Survey Manager Dataset Filter Creation Status

The 'CATI Survey Manager' in action: Each time the application is invoked from a 'new browser session', the caller receives the following screen:

CATI Survey Manager

What is the "Caller Number" you were assigned? 3
What is your FIRST name? Pati
How many "Call entries" would you like displayed per page? 5

<<Display Entries>>

CATI Survey Manager Initial Browser Session Configuration Screen

The caller enters the appropriate identification information and clicks 'Continue'. Now that the application has all necessary information, the caller is presented with the first screen of contact information.

CATI Survey Manager

(Call tracking information for Caller #3, Pati)

ID#	NAME	NAME	GENDER	RACE	SOC	ORGANIZATION	PHONE	DATE	TIME	CALLER	DATE	STATUS
254102097	Capt.	ABAR MICHAEL A II	M	C	USAF	69 FIGHTER SQ	4603169	460-4211	EST	**	31 Jul 2000, (Mon)	New
224079915	Capt.	ABEA DAVID W	M	C	USAF	71 FIGHTER SQ	5743500	574-1110	EST	**	31 Jul 2000, (Mon)	New
490888138	Capt.	ABEAS GAYLORD L	M	C	ROTC	91 AIR REFUELING SQ	9686485	968-1110	EST	**	31 Jul 2000, (Mon)	New
560640060	Capt.	ABDEEN FARLEY A	M	X	ROTC	510 FIGHTER SQ	3148801	314 632-1110	AFE	**	31 Jul 2000, (Mon)	New
589092787	Capt.	ABERNATHY ANTHONY D	M	C	ROTC	61 FIGHTER SQ	8965588	896-1110	MSI	**	31 Jul 2000, (Mon)	New

☐ Eastern	☐ Completed	☐ Officer	☐ Enlisted	☐ Male	Race
☐ Central	☐ Call Back	☐ 2Lt	☐ AB	☐ Female	SOC
☐ Mountain	☐ Retry Call	☐ 1Lt	☐ Ann		Organization
☐ Pacific	☐ TDY	☐ Capt	☐ A1C		N/A
☐ Hawaii/Alaska	☐ On Leave	☐ Maj	☐ SrA		N/A
☐ Europe	☐ Unavailable	☐ LtCol	☐ SSgt		N/A
☐ Japan/Korea	Name	☐ Col	☐ TSgt		N/A
☐ Middle East	SSN	☐ BG	☐ MSgt		N/A
	Caller #	☐ MG	☐ SMSgt		N/A
	<-- Date Filter -->	☐ LG	☐ CMSgt		N/A
List Multiple: 5		☐ Gen			N/A

<< Submit >> Set Filters Clear Filters Shut Down Stats << Submit >>

CATI Survey Manager Contact Information /Filtering Screen

There are two sections in the application's output display. The top 'contact' section contains the contact information required to place the calls, and the bottom 'filter' section provides the caller with the controls necessary to very selectively filter the contact items displayed.

The top 'Contact' section: The data displayed in the top section are all contained within the sample dataset. They contain the information necessary for callers placing the calls to gain the required understanding of the subject matter of the CATI conducted.

The data contained in the 'Name' field has a different appearance than the rest because it is a hyper-text link leading to a WEB-based survey. The link also includes information necessary for the analysts to track the demographic information of each respondent and is included in the response dataset.

The fields 'Caller', 'Date', and 'Status' are added onto the sample dataset during the initial configuration run of the application and enable the dataset to be used as a tracking mechanism. The 'Caller' field data is automatically entered for each observation where the 'Status' field has been modified and is based on the information provided by the caller on the initial run of the day. The date field keeps track of calls made to personnel that are not available but may be available within the data collection window. It allows for the selection of any weekday date during the next

13 days, and is used on subsequent days as a filter to target respondents. The 'Status' field enables the caller to indicate one of eight possible values for each call placed.

The bottom 'Filter' section: The filtering options are presented to the caller in two distinct regions. In the left region we observe the five 'default' filterable items: Time Zone, SSN, Name, Rank and Gender, corresponding to the 'required' data-items included in the sample dataset. Along with these we see filters for Caller # and Date. Together these comprise the 'static filters'. During the 'configuration phase', the analyst was requested to indicate which 'additional' data items should be included for filtering of contact observations. The right region contains these 'dynamic' filter items in the form of user selectable drop-down lists. This approach was chosen because of the ability to display a dataset containing data items without restriction. This capability can yield data elements/variables that contain literally hundreds of unique values and drop-down lists pose no display/selection restrictions.

All filters with the exception of 'Name' and 'SSN' are inclusive, meaning all selections are taken into consideration when filtering the data. To set, clear, or modify filtering combinations, the user simply selects the desired filtering options in combination with the appropriate function button on the display. 'Name' and 'SSN' searches are exclusive. Other filtering options are temporarily ignored, along with page display restrictions, to allow the display of all targeted individuals.

Once a caller has completed placing calls for all of the personnel listed, they click on the 'Submit' button to perform the required update to the status of the listed contacts. This action causes the application to call itself (recursion) and the process starts over.

UNDER THE HOOD

From the WEB to SAS®: Applications using SAS® Intrnet normally utilize the SAS® 'Broker'. The broker reads an HTML page, identifies the named items and their associated values, and then passes this information during its call to a SAS® application server. As part of the application server invocation, the broker passes along the information it has found in name=value pairs.

All applications using the broker must have the 'ENTRY' statement as its first line such as: '**entry optional = params &**'. This statement allows the WEB-based application to accept the incoming data stream passed by the broker in its call to the application. The 'params' argument in the entry statement above 'names' the SCL list built by SAS® when it encounters the statement. The power of SCL lists is harnessed by this type of application. Every item of data and information gathered and used by the application is encapsulated within an SCL list, providing the developer a single source for configuration, state and operation information, as well as the actual data provided by the users.

Selecting Filter Items: The code that allows the analyst to pick and choose the additional data items to be used as filters follows:

```
ds = open(DATASET,'U');
colcount = attrn(ds,'NVAR'); ← Get number of data items in the dataset
call set(ds);
count=0;
varname = makelist(); ← Create SCL Lists to store variable Name, Label and Length of data items
varlabel = makelist();
varlength = makelist();

*** Get Data Item Count & Load Names, Labels, and Lengths Into SCL Lists ***/

do index = 1 to colcount; ← Loop through the dataset and add Name, Label, and Length information to the 3 SCL Lists
  rc=INSERTC(varname, upcase(VARNAME(ds, index)), -1, upcase(VARNAME(ds, index)));
  rc=INSERTC(varlabel, upcase(VARLABEL(ds, index)), -1, upcase(VARLABEL(ds, index)));
  rc=INSERTC(varlength, VARLEN(ds, index), -1, VARLEN(ds, index));
end;
```

Creating Filter Items: The code that generates the data item filters follows:

```
ds = open(DATASET,'u');
COLCOUNT = attrn(ds,'NVAR');

FILTERCOUNT = 0;
do INDEX = 1 to COLCOUNT;
  rc=clearlist(VARVALS);
```

```

if nameditem(params, 'FILTER'||INDEX) then ← If the analyst indicated to use this data item as a filter, then create the filter
do;
  FILTERCOUNT = FILTERCOUNT +1;
  rc=varlevel(ds, vname(ds, INDEX), NLEVELS, VARVALS); ← Get unique values and place them in SCL List
  rc=sortlist(VARVALS); ← Sort the SCL List in ascending order for display (makes it easy for user to locate desired value)
  do x = 1 to listlen(VARVALS); ← Loop through the values for this data item to determine the longest one (used to set up output display)
    if length(trim(getitemc(VARVALS, X))) > MAXLENGTH then
      MAXLENGTH = length(trim(getitemc(VARVALS, X)));
  end;
  NEWLIST = vname(ds, INDEX);
  NEWNAME=NEWLIST;
  NEWLIST=copylist(VARVALS);
  FLTRVARS = insertl(FLTRVARS, NEWLIST, -1, NEWNAME); ← Insert the SCL List containing unique items into a master filter SCL List, by name
  VARLABS = insertc(VARLABS, varlabel(ds, INDEX), -1);
end;
end;

rc=sort(ds, 'AMF');
rc=close(ds);

FLTRVARS = insertl(FLTRVARS, VARLABS, -1, 'VARLABS');
FLTRVARS = insertc(FLTRVARS, MAXLENGTH, -1, 'MAXLENGTH'); ← Store the length of the longest unique item value
rc=savelist('catalog', 'CATIDATA.SLIST', 'INDATA' || '.slst', FLTRVARS); ← Save filters for subsequent calls

```

Creating Application State and Control Parameters: The recursive nature of this application requires several pieces of information to be present so that the application knows exactly what's going on at all times. The simplest way to do this is to transparently imbed the needed parameters into the actual HTML displayed to the user. This has the added benefit of providing an extremely dynamic nature to the application since the settings are localized. Here's how its done:

```

rc=fput(web, '<form method="POST" target="_parent" action="/vbin/SURVEYBROKER.exe?">');
rc=fput(web, '<input type="hidden" name="_debug" value="||DEBUG||">');
rc=fput(web, '<input type="hidden" name="_program" value="CATIPROG.programs.CATI.scl">');
rc=fput(web, '<input type="hidden" name="DATASET" value="||INDATA||">');
rc=fput(web, '<input type="hidden" name="NEWCALLER" value="||getnitemc(params, 'NEWCALLER')||">');
rc=fput(web, '<input type="hidden" name="NEWCALLERNAME" value="||getnitemc(params, 'NEWCALLERNAME')||">');
rc=fput(web, '<input type="hidden" name="LISTNUM1" value="||getnitemc(params, 'LISTNUM1')||">');

```

THE BENEFITS

The benefits of data-driven, WEB-based applications are astounding. They include:

1. **No software maintenance costs/time:** As the data the application must access changes, the application changes with it. The data independence coded within the application ensures this.
2. **Reduced cost in data access:** Using a WEB approach means that the organization no longer has to license a copy of SAS® for each workstation requiring access to SAS® data. (There goes the SUGI invite).
3. **Increased information availability:** This compliments the previous benefit. The WEB approach only requires the user have a WEB browser, access to the Inter/Intranet, and the SAS® Application Server. This is a tremendous benefit for those who have to go on the road.
4. **Centralized control of data:** All data is hosted and managed from one SAS® Application Server, this removes the problem of synchronizing data between systems. Everyone always reads from the 'same sheet of music'.
5. **Application Portability:** Transfer between platforms simply requires a re-compile of the source file using the SAS® system on the new host to make the application compatible with the new platform.
6. **Quick, Simple Recovery:** This type of application is easy to code and manage in a single file. Recovery then consists of recompiling a copy of the source file to bring the application back to an operational state.

CONCLUSION

The ever-changing information needs of organizations is rapidly requiring more and more applications capable of coping with these changing needs. The ability to create this type of dynamic application exists within the SAS®

System. By simply thinking in more generic terms when dealing with data and how we interact with it and applying a few SAS® functions, we can empower our applications to keep up with this dynamic world and meet the challenge!

Biography

Master Sergeant Bernard Poisson designs WEB-based survey applications for the AF Survey Branch. He created the systems specification for a survey targeting 530,000 personnel. His previous assignments include three years programming the radar systems AWACS aircraft, three years programming radar-modeling software used in testing FAA radar systems. Sergeant Poisson holds a B.G.S. degree majoring in Systems Analysis from Roosevelt University, Chicago, and an A.S. in Computer Science, from the Community College of the Air Force.

MSgt. Bernard R. Poisson
HQ AFPC/DPSAS
550 C. Street West, Suite #35
Randolph AFB, TX 78150-4378
(210) 565-2448
e-mail: bernard.poisson@afpc.randolph.af.mil