A Computer-Assisted-Telephone-Interviewing (CATI) system, designed and implemented by Atlantic Research Corporation, has been used for four years to conduct fast response surveys. This CATI system is an application of SAS Institute’s Full Software Product (SAS/FSP®) on a MicroVAX under VMS. The menu-driven system is programmed in the Digital Command Language (DCL) which invokes SAS routines to collect and edit the data, as well as produce standard reports and monitor system status. The system was designed to be user-friendly, reliable and have good screen response time. This paper will address the hardware configuration, the software implementation and techniques used to manage the survey.

During the last decade, Computer-Assisted-Telephone-Interviewing (CATI) systems have supplanted the traditional paper-and-pencil procedures for administering survey questionnaires by phone (Berry and O’Rourke, 1988). The purpose of this paper is to describe a SAS-based CATI developed by Atlantic Research Corporation (ARC). This CATI has been used successfully for the past four years to collect data from universities and other postsecondary institutions for the U.S. Department of Education’s National Center for Education Statistics. It has been run on both PC and MicroVAX platforms.

A CATI system is used for telephone interviewing and related forms of data collection such as data entry, editing, and coding (House and Morton, 1983). It consists of one or more workstations, each with a computer monitor and a telephone equipped with a hands-free headset. The CATI program produces a series of screens that contain the survey questions, space for the interviewer to enter appropriate data, processing instructions for the interviewer, and editing and coding routines, as appropriate. Each screen may contain a single question, a group of questions, or answer confidentiality rules for many items, often in tabular format (Nicholls, 1988). Additional screens provide the information necessary to contact the respondent (e.g., name and phone number), to record the results of a call to the respondent (e.g., wrong number, no answer), space to enter additional information necessary to manage the survey (e.g., call-back at a later time) and an outcome history screen to browse previous call events.

The most common use of a CATI is to collect survey data for projects such as political polling, market research, and government-sponsored surveys, e.g., the U.S. Bureau of the Census. However, the principles can be generalized to include many telephone-based data collection applications such as those performed by hotels, rental agencies, insurance companies, and airlines.

CATI systems have several advantages compared to paper-and-pencil systems. First, the use of a CATI system can reduce the amount of time necessary to complete a survey because it eliminates all data entry from paper forms and may reduce or eliminate the coding and editing functions as well. For example, edits may be included to ensure that data provided by a respondent do not fall outside a given range. Second, a CATI system permits projects to be more efficiently managed. All interviews may be conducted centrally which allows more careful management of interviewers by fewer supervisors. The results of each call and the data collected can be known immediately. This would allow the manager to know the direction of statistics being collected immediately. Finally, for very sophisticated surveys, particularly those with complex branching of questions, the quality of the data is generally higher for CATI surveys (Catlin and Ingram, 1988; Groves and Miller, 1987; Groves and Mathiowety, 1987; Harlow, Rossenthal, and Ziegler, 1988; House and Morton, 1983). However, CATI systems have two major disadvantages: time and cost. It takes longer to prepare a CATI-based survey than to prepare a paper-and-pencil survey. How much longer depends upon the complexity of the questionnaire and the amount of coding and editing required. In addition, CATI surveys are more costly because of the need to acquire computer hardware and to develop the necessary software. Furthermore, CATIs require computer programmers to develop and maintain them, thereby increasing personnel costs.

The question then becomes how to weigh the advantages versus the disadvantages. ARC decided to develop a CATI based upon the following considerations:

- Efficiency. We knew that CATIs were more efficient than paper-and-pencil surveys. Survey management resources were limited for our surveys and we believed that the CATI would help us maximize our limited resources (e.g., people, time).
- Long-term usage. Although it takes longer to prepare a CATI system, the surveys we planned were repetitive. Therefore, the same CATI hardware and software could be used, with only minor modifications, over a course of a five year contract.
- Cost. Similarly, the cost of preparing the system could be spread across five years. In addition, the same basic system could be generalized for similar surveys and the costs amortized across them. Because we had considerable expertise in SAS developing other software applications, we believed that we could reduce some of our costs by building the CATI using this language.

In addition to our SAS expertise, a MicroVAX II computer with twelve WYSE terminals were available, eliminating the need to purchase expensive hardware. The MicroVAX has a TK50 tape drive, two hard drives with 539 megabytes of storage, 16 megabytes of memory and runs version 4.4 of VMS. The WYSE terminals emulate VT220 terminals.

The only SAS software products that were utilized for our CATI were base SAS and SAS/FSP. The Digital Command Language, DCL, was used instead of SAS/AF®, which was not available in the VAX environment when the system was designed. The utilization of DCL allowed the system to be menu driven similar to SAS/AF. Examples of the menus are presented below. They allow the survey manager to run the entire system without any technical knowledge.

**THE ARTIST CATI**

1. Edit Main Database
2. Browse Main Database
3. Distribution Menu
4. Reports Menu
5. Shutdown System And Update Database
6. Show CATI Users and Space on VAX
7. Exit This Menu

Above is the first menu the survey manager will see upon logging to the system. The Main Menu is the access point to: 1) the other menus, 2) SAS programs to edit or browse the database, and 3) execution of DCL commands. If the survey manager selected option 3 from the above menu, the following Distribution Menu would appear.
DISTRIBUTION MENU
1. Station-1 with Call-Backs and up to 9 stations with default
2. Call-Backs and Appointments only distributed to up to 10
3. Station-1 with Call-Backs, Station-2 with Survey-Not-Available's, and up to 8 stations with the default
4. Station-1 with Call-Backs, Station-2 with Appointments and up to 8 stations with the default
5. Station-10 with Completes, Station-1 with Call-Backs, Station-2 with Survey-Not-Available's, and up to 7 stations with the default
6. Distribute the database into equal parts (At least 2 must be selected)
7. Return to Main Menu

* Default Distribution: Disperse all other Incompletes and non-refusals to a number specified by the manager.

The above distribution menu runs specific SAS programs to divide the database among the interviewers' stations as specified by the manager. If the manager would have chosen option 4 on the Main Menu, the following Reports Menu would appear.

REPORTS MENU
1. Item Non-Response Report
2. Out of Scope Reports
3. Response Rate Report
4. Interview Performance Report
5. Disposition Code Summary Report
6. Survey Not Available Report
7. Hard Refusal Report
8. Exceptions Report
9. Soil Refusal Report
10. Bitter End Report
11. Return to Main Menu

Even though the system is now quite elaborate, at conception it was simple and only required basic concepts. The following fundamental procedures are essential to any CATI system:

- Distribute the database to more than one interviewer (either a constant number or variable number). Since SAS/SHARE was not available, we emulated this networking procedure.
- Update the sample database with the new information collected during the session, where each session consists of a distribution and an update of the database; usually once per eight hour day.
- Produce a number of predefined reports to aid in the management of the survey and to create summary statistics and reports for clients.

SPECIFIC PROGRAMMING STEPS FOR THE ATLANTIC RESEARCH CORPORATION TELEPHONE INTERVIEWING SYSTEM (ARTIST)

1. Start by defining the population sample and constructing the SAS database. After this is complete, SAS/FSP can be used to customize the screens. In PROC FSEDIT, type MODIFY (or use the PF2 key) at the command line to enter the screen modification mode. Select option 2 and begin customizing the screens. When finished, type END at the command line. Protecting the screens is necessary to avoid having an interviewer modify any of the historical or sample information (i.e.: respondent's name, city or state). Range checking and setting protections on certain fields can be done by selecting option 4 (Field Attribute Mode). Under this option it is possible to set initial, minimum, and maximum values, as well as to set required fields and protection on fields. Refer to the SAS/FSP manual for help with range checks and other specific procedures.

2. At this point it is necessary to emulate SAS/SHARE® and give each interviewer a portion of the database from which to call and edit. The programs to distribute the sample to a number of interviewers can be as general as dividing the database in two or more equal parts or as specific as separating the appointments to one interviewer, the east coast respondents to a second interviewer, and the Spanish-speaking respondents to a third. It also is helpful for the survey manager to be able to vary the number of interviewers depending on the work load. Since the full SAS/SHARE language was not yet available in the VAX environment, a flat file was used to store this number selected by the manager. The flat file was then read into the distribution program to allow a variable number of distributions. This technique gave us some flexibility with how the database was to be distributed. The program code is presented below.

```sas
/* include ARTIST system formats */
libname forms ("ARTIST/ARTSTFORMS");
libname forms:
/* allocate datasets for master file and users */
filename stations ("ARTISTstas_max_user_for_current_cat_session.dat");
libname ARTIST1 ("ARTIST1. ARTIST88");
libname ARTIST2 ("ARTIST2. ARTIST88");
libname ARTIST3 ("ARTIST3. ARTIST88");
libname ARTIST4 ("ARTIST4. ARTIST88");
libname ARTIST5 ("ARTIST5. ARTIST88");
libname ARTIST6 ("ARTIST6. ARTIST88");
libname ARTIST7 ("ARTIST7. ARTIST88");
libname ARTIST8 ("ARTIST8. ARTIST88");
libname ARTIST9 ("ARTIST9. ARTIST88");
libname ARTIST10 ("ARTISTIC. ARTIST88");

data work.arnome:
set ARTIST1.master;
priority = petition.priority.;
timeout = (noncomplete=stintime, timeout); run;
proc sort data = work.arnome;
by priority timeout key;
run;
/* set up macro variables for each interviewer */
data_null;
infile stations:
input number length lastname $ 40;
do l = 1 to number;
call symput ('dsn'i'left(put(l,2.)) = output &lastname '.artnorm');
if l eq 1 then
lfast = 'I';
else if l eq 1 then
lfast = 'else if output = ';left(put(l,2.)) ;
else if l eq 1 then
lfast = 'outpoint = ';left(put(l,2.));
call symput ('ifstmt' = left(put(l,2.));
do i = 1 to number;
call symput ('ifstmt' = left(put(i,2.));
call symput ('ifstmt' = left(put(i,2.));
end;

data work.arnome:
set ARTIST1.master;
priority = petition.priority.;
timeout = (noncomplete=stintime, timeout); run;
/* evenly divide schools among users */
data adnum1
adnum2
dnum3
dnum4
dnum5
dnum6
dnum7
dnum8
dnum9
dnum10;
retain account 0;
set work.arnome nobs = master;
if g - 1 then account = master/nobs;account + 1;
else output adnaccount;
account count outpoint timeout priority;
run;
*/
At the conclusion of each session, the survey manager must include:

- posting the results of calls made during the previous session to the Call History Screen;
- machine editing newly collected data, and
- posting this edited data to the main database.

The posting of new data seemed simple enough at first glance: create a transaction file and use the UPDATE statement to update the database. One must consider two minor points for this procedure to work properly.

a. First, how many interviewers did you distribute calls to that session? Every session may have a variable number of interviewers working, but the maximum number of interviewers will generally be greater. For example, ARTIST has a maximum number of 10 workstations. It is not a good idea to use the maximum number of interviewers to update the database or old data could replace newly collected data. For example, if in the first session we distributed to 10 interviewers and in the second session we only distributed to 5 interviewers, the update in the second session would result in old data replacing newly collected data. This occurs because interviewers 6, 7, 8, 9, & 10 have the "old" session data which will write on top of the "new" session data for interviewers 1-5 if any interviewer from 6-10 has any new data. This code could wipe out previously collected data.

b. The UPDATE statement will not write a missing value over a non-missing value. For example, in a case where it is necessary to call a respondent back to verify over a non-missing value. For example, in a case where it is necessary to call a respondent back to verify data, the respondent could say they misunderstood the question previously and the question does not apply. The interviewer would be instructed to code an "a" in the cell where previously there had been data. Using the update statement would not replace the data with the missing value "a". The code below presents a technique for the use of formats to correct the missing data problem and the use of a flat file to store the session's interviewer count to correct the problem of writing over data incorrectly.

```
/* Include ARTIST system formats */
libname formats 'INTERNASTE.iformats';
libname formats.

/* Allocate the number of users brought up in the morning */
libname ARTIST 'INTERNASTE.source';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';
libname ARTIST 'INTERNASTE.ARTIST08';

/* Allocate datasets for master file and user */
libname ARTIST 'INTERNASTE.ARTIST01';

/* Note: No numeric data may have negative values */
proc format:
  value MISS
    A = '-1';
    B = '-2';
    C = '-3';
  end;

/* Special format to preserve missing data */
proc format:
  value MISS
    A = '-1';
    B = '-2';
    C = '-3';
  end;

/* Combine the schools */
data work.trans;
  set ARTIST1.arnonm
    ARTIST2.arnonm
    ARTIST3.arnonm
    ARTIST4.arnonm
    ARTIST5.arnonm
    ARTIST6.arnonm
    ARTIST7.arnonm
    ARTIST8.arnonm
    ARTIST9.arnonm
    ARTIST10.arnonm;
run;

/* Sort the combined records by key */
proc sort data=work.trans;
  by key;
run;

/* Update master file with combined records */
data ARTIST.master;
  update ARTIST.master
    work.trans;
  by key;
run;
```

4. A second menu should be created to generate any specific reports the manager will need. The manager can then have
In conclusion, our experience suggests that a highly efficient CATI System can be developed in a relatively short period of time, using basic SAS procedures. Furthermore, this system may be easily adapted to more than one specific questionnaire or data-gathering activity. Therefore, hardware, software and personnel cost can be spread across several different applications. A SAS software written CATI system can improve consistency, and monitoring capabilities. Thus, the necessity for any application compared to a paper-and-pencil implementation of Computer-Assisted and Hard Copy Telephone Interviewing." American Journal of Epidemiology 122: 335-349.


For more information, contact:
ARC Professional Services Group
Software Applications and Training Operation
601 Indiana Avenue, N.W., Suite 501
Washington, DC 20004
202/737-2666

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

BIBLIOGRAPHY
