Blaise CATI at Mathematica

Leonard Hart & Linda Bandeh, Mathematica Policy Research,. USA

1. Abstract

With more than 250 workstations at two locations, Mathematica Policy Research, Inc. (MPR) has one of North America's largest computer-assisted survey capabilities for primary data collection supporting public policy research. To insure that these survey operations run efficiently MPR has undertaken the development of reusable components for major aspects of the Blaise system. Three major parts of this overall capability are covered in this paper: (1) Blaise instrument shells for list, random digit dialing (RDD), and establishment surveys; (2) supervisor review utilities; and (3) overnight batch processes.

2. Introduction

Most survey organizations conduct surveys using some form of Computer Assisted Telephone Interviewing (CATI). CATI procedures, however, vary greatly from company to company, or even survey to survey within the same organization. Some organizations are able to use the built-in CATI features that come with their Computer Assisted Interviewing (CAI) package, while others have to build complex CATI systems to meet their needs. The approach an organization chooses depends on the level of management desired in managing a survey. In this paper, we describe how Mathematica Policy Research, Inc. (MPR) is adapting the standard Blaise suite CATI tools and how MPR has gone beyond these standard Blaise tools to do a CATI survey.

Blaise is a Commercial Off The Shelf (COTS) CAI package that can perform almost any survey needs an organization might need. Over the past couple of years, the survey community has seen great advances from Statistics Netherlands, they have added new features and extended the capabilities to the current Blaise suite. It is safe to say that no two survey organizations are the same; each has their own special needs. In the competitive contracting world of survey data collection, CATI management is the one capability that sets the various organizations apart.

For more than 20 years, MPR used a competing CAI package. During this time, the CATI management system at MPR developed into a highly complex management environment that met the needs of MPR clients. In the late 1990s, MPR began to see a change in clients' requests about how they wanted a survey managed. There were also changes in the technical environment and concerns about support and "product life" of the current CAI package. Clients started demanding a COTS package that would allow them to move data collection operations from one survey organization to another without having to do a major rewrite. After researching the different types of CAI packages on the market and after listening to client requests, MPR decided to move to Blaise in late 1999. Since that time, MPR invested in moving its CAI operations to the Blaise environment. MPR has completed four major surveys in Blaise. It has three major projects scheduled for early 2003, with more scheduled for later in the year.

Since MPR started using Blaise, we have learned much about the Blaise system, including some limitations. Blaise is a powerful COTS product that is hard to beat for easily getting a survey going with basic CATI management features. For some CATI management features, however, additional infrastructure is needed. This is

not due to a lack of support from Statistics Netherlands. As was mentioned, no two survey organizations manage a CATI survey exactly the same; each organization has its own specific needs. Because of this, it is not feasible for Statistics Netherlands to build a product that meets everyone's needs. To address this problem, Statistics Netherlands provides the user with a suite of tools that enable the company or organization to build, fairly easily, its own CATI management system.

This paper describes three key areas MPR felt additional features were needed to go beyond the basic CATI management facilities provided in Blaise in order to build a complex multimode CATI management system. These are: (1) Blaise instrument shells for list, RDD, and establishment surveys; (2) supervisory review utilities; and (3) overnight batch processing. One may think of these areas as the supporting legs that support CATI management. MPR calls the suite "MPR CAI Plus." This paper also touches briefly on a test environment that insures the quality and reliability of these pieces.

3. MPR Generic Blaise Instrument Shell

After fielding four projects with Blaise, MPR decided that it was the appropriate time to reevaluate the features available in Blaise, as well as the list of features that were needed in order to fielding CATI surveys cost effectively. Although the Blaise software is robust and has many features our previous CAI software did not, as with any COTS software, it did not cover all the features MPR desired. There were still areas that needed to be fine-tuned to meet MPR's needs and management style. After gaining some experience with Blaise, and developing an early version of a generic shell, MPR decided to put off additional Blaise surveys until we had had a chance to review what we learned, to write and update requirements, and to further develop our own "front-end shell." For the purposes of this paper, the generic shell is defined as a set of programs and processes that manage cases, that enable supervisors to review problem cases, apply interim and final status codes, and produce monitoring reports such as appointment and daybatch analysis reports.

Besides the features MPR wanted to add or change in the Blaise system, there were several primary goals for the new shell. We wanted to:

- Develop a flexible outcome-coding scheme that allows projects to add and customize codes to specific needs, while maintaining a standard set of codes that can be processed consistently across projects.

- Create a standard set of variables to be used for reporting and analyzing call attempts.

- Develop an interactive method of allowing project staff and programmers to select outcome codes and set related parameters without changing source code or recompiling the program.

- Allow the flexibility to change parameters for cases that need special handling, or for experimental design.

- Give supervisors more tools with which to review problem cases and monitor samples.

- Improve the history file display and access for interviewers and supervisors.

- Provide an easy-to-use parameter-driven system.

- Reduce programming and implementation costs.

To accomplish this, MPR organized a group of people, with representatives from each of the five main survey groups. Called the "Front-End Working Group" (FEWG), this group consisted of Karen Cybulski, a Survey Researcher; Barbara Carlson, a Senior Sampling Statistician; Larry Snell, Senior Manager of the Princeton Operations Center; Barbara Kolln, a Systems Analyst; and Linda Bandeh, Assistant Director of System Development, the chair of the FEWG.

The FEWG met once a week for a period of about eight months. The group wrote requirements without consideration for any particular platform. In other words, the intention was to not limit requirements according to what was available in either Blaise or our current software; instead, the requirements based on those that MPR staff felt were important and necessary. As the requirements were written, the Blaise development team reviewed them and asked questions. The team consisted of Carlo Caci (Programmer), Leonard Hart (Systems Analyst), Doug Dougherty (Senior Systems Analyst), and Mark Pierzchala (Senior Analyst for Computer Assisted Interviewing Methodology). As a result of this review, some specifications were modified in order to accommodate the existing limitations of the software, while other specifications were put on hold until the next version of the shell. Some of the goals are listed below.

3.1 Develop a Flexible Coding Scheme

The first requirement the FEWG tackled was to determine a flexible coding scheme for outcome variables. This requirement was, by far, the largest task; but, without it, many other tasks could not be started. To accomplish this, the group reviewed the status codes used on numerous projects over the past 10 years and borne out by the American Association of Opinion Research (AAPOR) scheme. First, the FEWG defined a structure of three-digit codes that would allow for expansion—for example, 200-299 would be used for refusal codes. Once this was done, the FEWG defined how this group of codes would be treated. Every code within this group would have a general meaning and a method of processing. For example, cases with refusal outcomes would be part of a refusal group controlled by number of refusal calls, days between refusal calls, and refusal letters, which would be assigned to refusal interviewers. This reduced the handling of specific codes and allowed new refusal codes to be added with minimal programming; for some processes, no additional programming would be required. The major outcome groups included completes, locating, refusals, contact calls (including appointments), no-contact calls, barriers, and phone problems. The most common outcomes were given a code that could not be changed; codes were grouped by type and function within the major groups and gaps were left between codes for projects to define any additional codes necessary.

Once the outcome codes had been defined, the FEWG examined each code and discussed how it should be processed, how and if the case should be delivered for the next and future calls, and how it should be reported.

3.2 Define a standard set of variables

The FEWG also defined the related variables that would be needed to control or monitor the case. As an example, an eligibility code—denoting eligibility for inclusion in relevant study—was removed from the outcome codes and a separate set of variables developed. These variables can be customized for a project's needs—thus allowing projects to define and track multiple levels of eligibility

without having to create an extensive list of outcome codes to cover all situations. For example, on an RDD study, there may be several levels of edibility, such as working phone, residence and household members over 18. Standardized reports are then developed that use the eligibility variables, along with outcome variables.

3.3 Develop an interactive method of selecting outcome codes and setting related parameters

MPR wanted an easy way for survey staff and programmers to select outcome codes and options from this list in order which to customize the shell according to their project's needs. The Blaise development team designed an interactive process that allows users essentially to go through a series of screens and pick or change options. The screen was set up with the recommended default options, but users can select outcome codes and change such parameters as: the number of days to hold the case before calling back after a refusal; the number of refusals to allow; whether an outcome should be final status or sent to a supervisor for review; the number of times to call before leaving a message; the number of days to hold a case between leaving messages; the number of phone problems before sending a case to locating; and so on. These options are implemented as external Blaise files, one for status codes and one for other survey settings

A Blaise shell was developed that reads and is driven by these parameter files. The external parameter files accommodate experimental design. There can be an external parameter record for each treatment, thus allowing projects handle cases differently depending on their experimental design.

3.4 Implement a set of supervisory tools

To define a set of tools and screens for supervisors, MPR reviewed the tools available in the Blaise system, requests from phone center supervisors, and feedback from other groups. MPR's system allows supervisors to review individual cases, groups of cases, final and interim status cases, history, all counters, dates, parameters, flags, and outcome related to the cases, as well as change those parameters, if necessary. (See figure 1 below.)

3.5 Maintain history files

Supervisors and interviewers requested a more extensive history than that which is currently provided by the Blaise system. They wanted a record of every call to be available on a screen, which would allow them to scroll through calls and see the time, date, interviewer, outcome code, appointment information, and notes from interviewers.

Using the native history file would not accommodate the needs of the supervisor or interviewer. To overcome this problem, the FEWG, working with their users, decided which key information would be maintained in the data model as part of the shell. The variables were then added to a block called "history" which was then arrayed for up to 100 calls. Next, code was added to all possible exits to the main shell, that would update this array each time a person or process touched a case.

3.6 Maintain adequate documentation

Writing and maintaining the requirements document was an important responsibility of the FEWG. Each member was responsible for representing his or her group's interests. During the requirement-gathering phase, several meetings were held. The staff were given copies of the requirements and invited to discuss them and ask questions. By having a representative group and conducting company-wide meetings, the group was able to ensure that all interests were

addressed. The requirements document became a crucial document for programming, testing, and documenting the new system.

After the review of the documentation by the Blaise development team, the requirement documentation was updated and sent back to the FEWG for review and comments. This process continued until FEWG had completed the requirements for version 1.0, and for several months into testing, to ensure that requirements had been interpreted correctly and that questions from the developers had been answered.

3.7 FEWG products

The FEWG outlined the requirements for calling routines and developed a set of standard outcome codes, counters, flags, timing variables, and history requirements. They not only looked at the information needed to conduct the survey, but at the requirements for data collected for reporting, monitoring, and analysis. When the FEWG was first formed, the intention was to write requirements for List, RDD and Establishment surveys. The focus was on developing a flexible architecture that would allow for other types and modes of surveying, and to develop a source code that was independent of project-specific screening and contact blocks. It quickly became clear that this was too large a task. List surveys are the most common types of surveys for MPR and the first project that was targeted to use the next version of the shell was a list survey.

The shell was developed so that the contact section, the introduction and the screens are separate blocks from the rest of the standard code. This allows projects to customize the screener and introduction to their project needs, in addition to making it the starting point for RDD or Establishment shells. While there will be different modules for the three shells, they will share much of the same source code.

4. Supervisor Utilities

Once the new version of the list shell was developed and was in the testing phase, the emphasis switched to developing supervisory tools. Blaise CATI management tools are probably the weakest link in the Blaise suite. The suite does come with a few easy-to-use utilities—for example, CATI Specifications, CATI Management, and CATI History Viewer or built in features like Time Slices, Time Zones and Groups. These, however, are insufficient for most survey organizations. As mentioned in the introduction above, every survey organization will have its own special need for tools with which to manage a survey, and it would be impossible for Statistics Netherlands to provide a single product that would accomplish all these tasks.

Blaise suite does provide other tools with which to write individual, in-house applications. In the past, one could use Manipula or Maniplus or a dll link into the DEP or Manipula/Maniplus through a Delphi hook provide by Statistics Netherlands. But, in the past few years, Statistics Netherlands has put great effort into developing the Blaise Component Pack (BCP). The BCP, a collection of several COM components written specifically for Blaise, can be used in applications or programming languages, so that the organizations can create the user's own customized tools for integration into the Blaise CATI management.

MPR Supervisor Utilities are part of a package called "MPR CAI Plus." MPR CAI Plus is built on the philosophy of extending the current Blaise suite of utilities to meet the needs of MPR and its clients. MPR CAI Plus is a generic system built by using Visual Basic, Manipula, Maniplus, Blaise suite programs, and third-party

applications. This package provides users with an easy-to-use menu system, control over granting access to certain tools, and various utilities for managing a multimode survey.

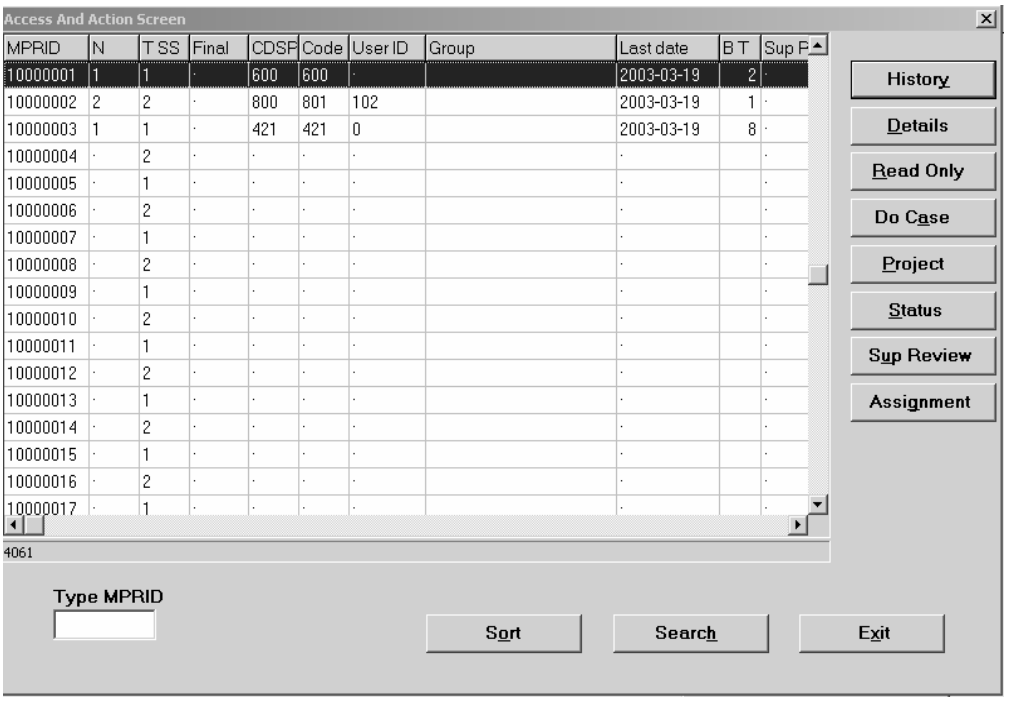
The first part of the Supervisor Utilities is the MPR CAI Plus menu system. This is a generic, easy-to-use menu system built by MPR with built-in survey management tools. A key feature of this package is the ability to set up a new survey and grant Supervisor or administrative privileges to certain utilities. As long as key survey files reside in the right place, a person can have a new survey setup and be ready to go within a few minutes at one of the call centers. In addition to these features, there will be reports that are ready to run, as well as case management tools for reviewing case progress.

The other part of the Supervisor Utilities consists of the generic Maniplus applications. These standard Maniplus programs are written to use the standard shell, with only a few minor changes needed for the Maniplus code; basically, for any survey, changing a few uses sections, with input or output statements pointing to the right folder structure, and these programs will fit right in with the MPR CAI Plus menu system

Figure 1 is a screen shot of one of the many Supervisor Review programs called "Access and Action Screen" that are part of the MPR CAI Plus system. For this particular Maniplus program, a CATI Supervisor has control over a case or a series of cases, review case information, set flags; it can even override certain flags. From here, it can sort, search, or go retrieve a particular ID. Also, once an ID has been selected, it can change a case, or "re-status" it, see predefined case details, or see the case history on a user-friendlier screen.

Several other prepackaged Maniplus programs round out MPR CAI Plus—for instance, a better Daybatch viewer, or a detailed appointment viewer or mark cases for review. This series of Maniplus programs, which come as part of the generic MPR CAI Plus suite, can, with little modification, be used for any survey. All the survey programmer needs to do is make a slight change, if needed, to the Maniplus program, re-prepare, test the finished MSU file in the test environment, then move the new MSU file into the live production. The MPR CAI Plus menu will automatically add it to the supervisor menu.

Figure 4 - Access and Action Screen



5. Overnight process

No CATI system would be complete without an overnight process. The user might want to process cases each night—assign treatment groups to cases, re-status cases, reset missed appointments, apply new rules, run reports, back up key data files, do maintenance of files or update outside data sets, and so on.

Like the other two pieces of MPR CATI management mentioned above, the overnight process was built on the foundation of a standard Blaise shell and Supervisor Utilities, to work efficiently. Like the Supervisor Utilities, the overnight process is a highly generic system that can be up and running within a few minutes. This generic process backs up key files multiple times throughout the process. It can tie into the Survey Management System (SMS) built with Microsoft SQL Server, run maintenance on files, and create/update files that are used by the Supervisor Utilities.

Currently, MPR feels that data manipulation can better be done as an overnight process that does not affect live calling. Over time, we feel that applications will become faster, hardware will get faster and cheaper, and software applications to do perform these tasks as real-time process will all become cheaper and better. MPR will continue to look at the options to do perform these tasks in real time as they become available.

6. Testing environment

As MPR moved from its old CAI package to Blaise, MPR realized that a testing environment closely resembling the live environment was a very important part of the move to Blaise. This test environment played an important role in building a generic CATI system. This environment became a proving ground for determining whether various features would work.

MPR sees this testing as an important part of CATI management. We feel that every effort should be made to replicate the production environment for this testing. Local machine testing is not adequate since there are so many variables that can affect how things will operate: different servers, the operating system, the wrong version of a dll, a network switch, and so forth. The purpose is to minimize the changes in moving from the test to the production environment.

MPR has tried to overcome this problem by replicating its test environment using the exact hardware, software, switches, and client machines. The test environment uses the same equipment and software as does production; it just maps to a different network folder. By doing this, we have caught things that went unnoticed when testing on local machines. We have also found differences in equipment and software that we might not have found until we moved into production. A good test environment is well worth its cost.

7. Summary

Over the past two years, MPR has invested heavily in laying down a sound foundation for a new CATI system. Many people have been involved, with many hours devoted to writing specifications and coding generic systems, investing in new equipment and software, and retraining people at all levels of the company. We feel that we are beginning to see this investment pay off. We can now get a new Blaise survey into production in very little time, and people have the utilities to manage their CATI surveys. We continue to look at new ways to improve our Blaise components and to integrate them into out outside applications. While no Commercial Off The Shelf software can meet every need, Blaise does provide the tools with which to adapt Blaise CATI to individual needs, just as MPR has done.