STANDARD MULTI-SURVEY SHELLS IN NASS

Roger Schou and Mark Pierzchala
National Agricultural Statistics Service, USA

1. Background

A "shell" refers to standard Blaise programs containing beginning and ending administrative modules that surround survey-specific questions. Three related shells will be used in almost all surveys conducted in Blaise in the National Agricultural Statistics Service (NASS). These are the List Frame Shell, the Area Frame Shell, and the Multiple Frame Shell. The List Frame Shell and the Multiple Frame Shell are now complete. The Area Frame Shell will be adapted from the Multiple Frame Shell by early 1994. In order to understand why these shells are important to NASS and why they are so complex, it is necessary to know a few things about the sampling schemes of NASS and the basic survey operations including interactive survey processing on NASS's many Local Area Networks (LANs).

NASS has 45 state offices, headquarters in Washington, DC, and a research facility in Fairfax, Virginia. This means that there are potentially 47 programming sites and 45 end-user sites for various national-level and state-level Blaise applications. Regardless of site, NASS employs standard sampling frames, sampling schemes, and survey procedures. This ensures that data collected in each state can be aggregated to national-level estimates. The shells embody standard NASS methodology for almost all surveys. By developing and distributing the shells throughout the agency, programmers can concentrate on just the survey-specific questions thus avoiding much duplication of effort. The shells ensure that surveys are conducted in approved ways without unduly limiting flexibility in each state office.

The complexity of the shells derives in part from the use of two conceptually different sample frames, the List Frame and the Area Frame. The sampling schemes based on the two frames require that sample weights take into consideration not only sampling rates within strata but also the degree of overlap between the list and area frames and of the duplication of potential respondents in the list frame. These considerations require many detailed and related questions about each individual associated with the farm or piece of land in order to determine who operates it. Farming operations can have many different types of ownership and operating arrangements including individual ownership, partnership, and corporate management.

Another major contribution to the complexity of the shells stems from the many tasks the Blaise system is being asked to do from one source code including interactive editing, Computer Assisted Telephone Interviewing (CATI), Computer Assisted Personal Interviewing (CAPI), survey management including call scheduling, and some data entry from paper forms. The fact that Blaise will handle multiple tasks is further complicated by the multi-mode nature of data collection in NASS. For most of the main agricultural surveys, data are collected both on paper and by computer assisted methods, primarily CATI. However, NASS will soon start a pilot program where data are also collected on laptop either in person (CAPI) or by phone from field enumerators' phones (decentralized CATI).When data are entered from paper forms, NASS presently uses a high-speed data entry package called Key Entry 3 employing item code data entry. This means that only data with item codes are keyed. In NASS questionnaires, many of the administrative and screening questions do not have item codes, yet they are essential to the flow of both the paper and the computer assisted interview. However, they are not needed for summarization and not keyed when the data arrive on paper. Since these questions are not keyed, much of the data from paper forms would be off the route if directly entered into an instrument that is constructed solely for interviewing. Therefore, the shells must take into account the source of the data and adjust the route accordingly.

There are other major differences between computer assisted interviewing and interactive editing in NASS. For example, an interviewer may be allowed to enter a DontKnow or Refusal for an item during an interview, but this is something that must be called to the attention of the data editor in order to personally impute a plausible value. Some edits may be soft in the interview but hard in editing. It is NASS policy to require each appropriate question to be asked in computer assisted interviewing requiring an answer of at least zero for many questions. On the other hand, during high-speed item code data entry, zeroes are not entered. Thus, a way must be provided to allow empty answers in editing while requiring at least zero answers in interviewing.

NASS makes maximum use of Blaise's processing modes, CADI for editing, CATI and CAPI for interviewing, and the ability to condition routes and edits on these modes. NASS also needs to condition routes and edits on survey task, for example data entry or verification.

Table 1. Processing scenarios

Processing scenario	**Remarks**
KE3 - IE	Data questions only on the route. Update call scheduler and NASS survey management system.
BDE - IE	Data questions only on the route. Update call scheduler and NASS survey management system.
CAI - IE	Keep all modules on route but invoke IE-specific tasks when appropriate. Update call scheduler and NASS survey management system.
CAI - CAI - IE	Update call scheduler and NASS survey management system when an interview is interrupted.
CAI - IE - CAI - IE	Keep all modules on route but don't undo any IE-specific work when going back to CAI. Update call scheduler and NASS survey management system.
IE - CAI - IE	Put all CAI modules on route when necessary. Update call scheduler and NASS survey management system.
CATI - CAPI - IE	Update call scheduler, NASS survey management system, and CAPI management system.

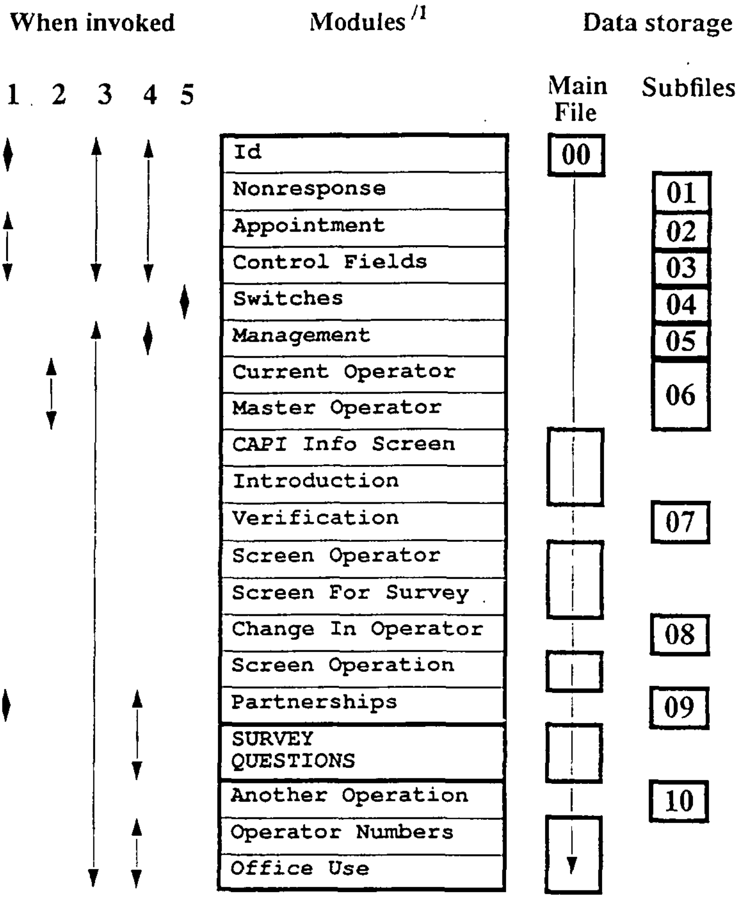
The shells provide hooks into NASS's other major data processing systems, for example with the sampling frames and a survey management system. In order to allow maximum flexibility for each state office, the shells must be able to handle each form individually in each of the scenarios in table 1. In the table, CAI = Computer Assisted Interviewing, IE = Interactive Edit, KE3 = Data Entry in Key Entry 3, and BDE = Blaise Data Entry.

2. Design of the Multiple Frame Shell

The shells have been developed to accommodate the specifications that have been generally explained above. They have been built in modules. The modular approach was adopted in the Multiple Frame Shell in order to facilitate maintenance, modification to other shells, and as a basis for the subfile layout. The Multiple Frame Shell and the List Frame Shell are related in that they both are built of many of the same modules. Generally, the List Frame Shell uses a subset of the Multiple Frame Shell's modules and altered versions of some of the others. Thus if a common module must be modified, it has to be modified in only one place. The modules are stored in a formal library of Blaise Code that is being built up to facilitate the production of instruments. Figure 1 presents the general structure of the Multiple Frame Shell, when each module is invoked for a few of the processing scenarios above, and where data are stored.

Extensive use is made of the Blaise subfile facility in order to ease pre- and post-survey processing of administrative data. For example, name and address updates appear in subfile 07 and do not have to be separated later from the data to be summarized. In addition, each subfile can be processed on its own schedule because Blaise keeps status information for each subfile separately. For example if data are read out from several subfiles but then data are changed in one subfile only, then it is possible to read out only the data that have been changed, by subfile. The use of subfiles also keeps data from being inadvertently overwritten. During initialization, certain control data are stored in subfile 03. When survey data are then read into the main data file, the control data will not be overwritten with blank values as

Figure 1: Schematic of Multiple Frame Shell



1 Data are read in during initialization
2 Data are computed during initialization
3 Invoked in CAI - IE scenarios
4 Invoked in IE - only scenarios
5 Computed during data file movement

1/ See Table 2 for an explanation of the modules.

they would be if they were also stored in the main data file. Finally, the use of subfiles saves a great deal of disk space. For example, the partnership information involves numerous questions. Since these are stored in a subfile, no storage space is taken up for farms that do not have partners. Only a small percentage of farms do have partners. In table 2 below, is a listing of each major module in the Multiple Frame Shell and its function.

Table 2. Function of each module

Module	**Function**
Id	Identification and administrative questions as well as questions used to flag forms for special handling.
Nonresponse	Survey-specific module containing global questions asked as a last resort in the event of a refusal. Automatically codes some administrative questions.
Appointment	Standard Blaise appointment block plus NASS additions that allow CATI/CAPI compatibility and hooks into NASS's other survey administrative systems such as the Name and Address file and the operator comment file.
Control Fields	Holding place for up to 10 sample control variables as might be required in a particular survey.
Switches	Processing switches (and selectors) for KE3 data entry, forms check in, and the Blaise utility Forms Manager.
Management	NASS-specific survey management questions are set here in order to define survey management reports once that apply to all surveys. A Pascal sub-routine reads DOS environment variables for the user of the system.
Current Operator	Name and address of the person **currently** considered to be the operator. Master OperatorName and address of the person thought to be the operator **at the time of initialization.**
CAPI Info Screen	Displays comments about the operator/operation if the CAPI mode is used (and hence the CATI call scheduler is not used).

Introduction	Introductory questions for interviewers, determines respondent, his/her relationship to the operator, and follow-up questions for out-of-business operations.
Verification	Name and address verification and updates for the operator and operation.
Screen Operator	Checks to see if any operators of an operation have been added or dropped.
Screen for Survey	Survey-specific screening for commodities of interest.
Change in Operator	Invoked when an operator is out of business. Tries to find the new operator and decides whether a new form is needed for substitution based on screening.
Screen Operation	Determines type of operation (individual, partnership, managed), and asks follow-up questions if necessary.
Partnerships	Name and address verification of up to 6 partners, determines which partner is the operator. Allows addition and deletion of partners.
Another Operation	Determines if the operator is responsible for another operation.
Operator Numbers	Gathers sensitive operator/operation numbers such as Social Security Number and Employer Identification Number.
Office Use	Administrative data entered only in the clerical or statistician edit.

3. Other aspects of the Multiple Frame Shell

The complexity of the Multiple Frame Shell is very difficult to convey in this short article; however, one indication of this is its size. When compiled without any additional survey-specific questions the shell contains 428 uniquely defined and 745 overall used questions in addition to several hundred computations and edits. For many of NASS's surveys, the number of questions in the shell will outnumber the number of survey-specific questions. Many of the modules are intricately interconnected. For example, when a change in operator is discovered appropriate changes have to be reflected in several different subfiles. This way the question fills for the interviewer will always be up-to-date and the call scheduler will bring the corrected names and addresses onto the screen in the case of an interrupted interview. The shell is also designed to be very robust. This means that considerable change can be made during or between surveys without disrupting calling or editing.

In addition to the Blaise code, a "protocol" around each shell has also been developed, which is part of any NASS instrument. In NASS, the term protocol refers to the subsidiary programs which surround survey processing. These include programs which transfer names and addresses from the frames into Blaise and back again if there are changes, electronic movement of forms from one Blaise data set to another, generation of survey management reports, the formatting of external files holding historic data, etc. Almost all of the protocol has been developed in Blaise utilities, especially Manipula (file manipulation and reports), Abacus (tabulation), and Forms Manager (movement between Blaise data sets).

4. Development experience and improvements in Blaise

The development of the shells has proved to be an iterative experience as NASS works to polish procedures for interactive processing with multiple modes of data collection using one source code and the same data sets. One of the most difficult aspects of development has been to find out exactly what needs to be programmed. NASS is just moving to a new screening algorithm which while simpler than the old one, still requires quite detailed programming of many different rules while accounting for many exceptions to the rules. In order to learn all of the procedures, many people were interviewed and the shells have passed through many reviews.

An early version of the List Frame Shell has been in operation for two quarters in the Catfish survey. The 15 state offices that conduct this survey have provided valuable feedback on the performance of the shell. This shell has performed adequately in this survey with one major omission being that when data were read in from Key Entry. 3 the call scheduler was not updated properly. Thus sometimes a call would be placed to someone who had already sent in a questionnaire. This omission has been taken care of in the current versions of the shells.

The Multiple Frame Shell was modified in a few days to serve as an ad hoc Area Frame Shell for the 1993 June Area Frame Survey research project. Some code was commented out and a few modules were added, but yet it was far easier to modify the already existing shell than to start from scratch. The changes that were made took into account the conceptual differences between the List and Area Frame sampling as done in NASS; however, the majority of the code that was kept ensured that the mechanics of screening (i.e., partnership verification) was done the same way between the two types of surveys. The Area Frame Shell will be formalized and more rigorously tested in early 1994 based on the experience gained in 1993.A few of the difficulties in programming the shells can be traced to the Blaise language and the Blaise way of doing things. As a result of the June Area Frame project, NASS identified 7 specific language enhancements that would ease multi-mode and multi-task instrument development. These suggestions have been forwarded to the Netherlands CBS and it appears that most or all of the suggestions will be implemented in Blaise, version III.

5. September Quarterly Agricultural Survey

The first operational test of the Multiple Frame Shell will have been conducted in September 1993 in three state offices. A companion paper by Asa Manning describes the planning that has gone into this trial. NASS fully expects to revise both the Multiple Frame Shell and the office procedures as a result of the September trial and then to test the revisions in December in the same state offices. The development of a processing flow in the multi-mode and multi-task environment has been itself iterative. Four different flows have been devised, each of which would work, but each having strengths and weaknesses. The fourth is a combination of two earlier flows. One of the biggest concerns has been the potential for interruption of some processes in order to perform other tasks. For example, if interviewers and editors are working off of the same data set, then the CATI would have to stop in order to perform a batch edit. In consultation with the Netherlands CBS, NASS has devised a scenario that should much reduce or eliminate tasks interfering with each other.