Centralizing the Mink Survey at the National Agricultural Statistics Service

Roger Schou and Emily Caron, National Agricultural Statistics Service, USA

1. Introduction

The National Agricultural Statistics Service (NASS) is an agency of the United States Department of Agriculture. NASS is responsible for collecting, editing, and summarizing agriculture data. We are the sole agency for producing Agriculture Statistics for the United States.

NASS has had the unique challenge over the years of disseminating Blaise instruments across forty-four field offices from our headquarters (HQ) location in Washington, DC. With the introduction of Blaise 4.81, we could begin the process of centralizing our instruments in order to create a seamless approach to CATI and paper data collection and editing across the U.S. The first survey to undergo this transformation was the Annual Mink Survey. But before we delve into the details of centralizing this survey, let's provide a little background about how we have handled Blaise surveys in the past.

2. The Old Days

Our older LAN-based Blaise instruments are invoked with a Visual Basic 6 menu system comprised of numerous builds. Whenever a new instrument is created or an update is needed for an instrument or the menus, a LAN Update is sent out to each affected field office (FO). LAN Updates generally consist of zipped up files and a .bat file that, when run, will save the new files to the FO LANs. Someone with administrative rights needs to be on hand to run these programs. Between the preparation, dissemination, and the running of the LAN Updates at the FO level, much time and effort is spent on this process. It can sometimes be the case that LAN Updates are run out of order which, depending on the nature of the update, can cause problems.

Current LAN-based Blaise instruments at NASS use Blaise 4.80 and write to Blaise datasets. Since these datasets are located on each FO LAN, HQ survey administrators are not easily able to view the progress of a given survey. In order for collected data to be analyzed or summarized at the national level, it needs to be read out of the Blaise dataset and sent by some means to a common location.

Over the years, NASS has expanded six of its FOs to act as Data Collection Centers (DCC). An FO might require assistance with phoning due to having a particularly large sample or a small or even non-existent phone interviewer staff, in which case they will put in a request to become a Client State (CS). A DCC will be assigned to them and will take over the responsibility of phoning the CS's sample. Working in a decentralized environment means moving zipped up datasets between CSs and the DCCs and depending on communication links being up in order to do so. Some of the surveys we conduct also make use of Estimation Centers (EC) which requires more data movement.

Considering the need for real time national status reports, seamless data transfer and the agency's overall direction toward enterprise architecture, it was time to make a change.

3. Centralized Mink Basics

In April 2010, NASS conducted its first survey in a centralized database. We chose the Annual Mink Survey because of its small sample size (only 415), relatively long data collection time frame and simplistic nature. It was also a good candidate because the data is collected and edited in only two states, Wisconsin and Utah.

We started out using Blaise 4.82 build 1529 and stored the data directly into a MySQL database. In order to maintain a common structure across all future centralized Blaise instruments, we used Generic BOI files. The in-depth data partition type was selected and versioning was turned on.

4. Updated CASIC Menu and System Requirements

We created a Web-enabled menu in Visual Basic .NET which is used to invoke Blaise and Manipula and manage the survey. Whereas our old VB6 menu was made up of numerous builds all working together, the new menu is one build and can be updated on the fly without needing LAN Updates.

Certain settings are needed on user workstations in order to run the new CASIC menu and centralized Blaise. Both the .NET Framework and MySQL driver must be installed. Eight Blaise DLLs must be registered and an oledb.dbi file saved on the workstation. In addition, ODBC System data sources need to be set up to provide connection information to MySQL. All of these system requirements were pushed out to workstations in the FOs involved in the Mink Survey prior to the survey start date.

5. User Access

One of the first things the VB.NET code checks when a user attempts to access the new CASIC Menu is whether or not the user has been granted access to it. We have accomplished this by assigning certain fields in the Windows AD table for valid users. We are able to tell where they are located (FO Fips or HQ), what their username and employee number are, and what role they have in the office (Statistician, Stat Assistant, Supervisory Interviewer, Interviewer). As long as the corresponding AD fields have valid entries, the user is granted access to the menu.

6. CASIC Tables

In addition to the eight standard Blaise tables that accompany Generic BOI files, we found the need to create a few others: CASIC_SurveyInfo, CASIC_FAT and CASIC_Management.

The CASIC_SurveyInfo table is one of the first tables populated for a given survey. This table holds critical information about each survey, such as survey code; year, month and day values; Blaise instrument name; different survey type indicators; data collection dates and other useful information. This table allows us to limit which surveys are visible to the user on our survey selection screen. Many values from this table are passed as parameters into our menu and Blaise instruments so we can have each of them react as needed.

The “FAT” in CASIC_FAT stands for FIPS Assignment Table. An interface was created within the new CASIC Menu to allow HQ staff to interactively assign CSs to DCCs and ECs for each survey. Once these assignments are created, the resulting data is populated into the CASIC_FAT table and the menu will know which rights to provide to each FO. So if state A is having their data collected by state B (one of the DCCs) and state C (an EC) is editing all of the data, the menu will react accordingly. Users in state B will have access to menu buttons related to the call scheduler and data collection. Users in state C will

have menu buttons connected to edit functionality. State A users (CS) will have a very small collection of buttons in order to prepare their sample for initialize, read documentation and run certain reports. There was a total of 21 FOs involved in the Mink Survey. Of that number, two played the role of both DCC and EC which left 19 in the role of CS.

Originally we did not have the CASIC_Management table, but quickly realized it was a necessity. This is a flat table containing many key Blaise fields on which we often need to sort or limit the datasets. These fields are defined as indexes and used in RecordFilters in Manipula. Some examples of fields found in this table are DCC_Fips, EC_Fips and Batch.

7. Directory Structure

The new structure for centralized surveys involves web, application and data servers. There are three environments to each: Development, Beta and Production. CASIC Menu code is located on the web servers, Blaise code resides on the application servers, and the MySQL databases are on the data servers.

We went back and forth on how to set up the directory structure for Blaise code on the application servers. By the time the survey went live, we had determined we needed specific folders designated for any FO playing the role of DCC or EC. In the case of Mink, that meant special directories for Utah and Wisconsin which would house state-specific files such as CATI Specs, .IGL and those related to the interviewer practice dataset (which we kept as a Blaise dataset). Since we found the instrument .BMI file needs to be collocated with the .BTR, we also have copies of the instrument files in the FO folders.

A directory designated for HQ was created, as well. The purpose of this area is to gather files that FOs have prepared for initialize (explained later), back up those files once they are processed, and to gather any special reports or message files that are generated after HQ processes run.

The one external used for the Mink Survey contains the sample master and we decided to keep it as a Blaise dataset. The external datamodel was required to be in the same folder as the instrument.

8. Division of Tasks

In older decentralized Blaise instruments, each FO involved in the survey would receive the instrument from HQ and do all of the steps needed for sample initialize. In the new centralized Blaise, getting the Blaise instrument ready to go requires actions from both the FO and HQ. Each FO that makes up the survey sample will need to run what we call Initialize Preparation. This is a process that ensures all pertinent files are in place and runs numerous checks against the FO's sample to be sure it's ready for initialize. If there are any problems found, a report will be presented to the user so they can take action. If no major problems are found, that FO's sample will be queued up for initialize. The Mink sample consisted of records from 21 different FOs, so each one of those states ran Initialize Preparation over the course of a week.

As FOs were lining up, we ran a manual process in HQ to initialize the samples into the dataset. Utah and Wisconsin were able to begin collecting and editing data as soon as at least one of their Client States' samples was initialized. Having a centralized dataset gave those two main players the ability to run important reports for their regions that were required to complete the survey, which was a big improvement over the past method of manually sending the reports between FOs.

Special reports were created for HQ including a couple of different status reports, missing reports and an initialize status report. Data readout for Mink was done at the HQ level on a daily basis. Records

considered to be complete and clean from across the entire sample were read out to an ASCII file, which was then picked up by the analysis and summary tool.

9. Issues We Encountered and Lessons Learned

Throughout data collection and editing, users reported CATI Service errors on a very regular basis. While researching the problem, we were able to get them back up and running by simply downing the service and bringing it back up but that, of course, was not a permanent solution. After speaking with database experts in our agency and getting the advice of Statistics Netherlands staff, we changed some of the ODBC settings on the different application servers. Coupling that with an updated Blaise software version drastically reduced the number of CATI Service errors for the remainder of the survey period.

The performance (speed) of the Mink instrument as reported by FO users was not ideal. Originally, the Blaise software was being accessed by the menu from the central application server. Partway through the survey we decided to instead hit the software from the local FO LAN, and this helped to improve performance. This was around the same time that we received an updated version of the software, which also helped.

10. Future Plans

After wrapping up the Mink Survey, we knew we had lots more work to do before tackling a much more complex survey. We also wanted to be sure a proper database failover was in place which was not the case during Mink data collection.

Knowing that speed and performance would only decrease as samples and instrument complexity increased caused us some worry. However, our agency has plans to convert all FOs to a virtual environment by the summer of 2011. Preliminary tests of a virtualized FO using centralized Blaise have looked promising, so we have high hopes that performance issues will be a thing of the past. Virtualization should also eliminate the need for specialized workstation installs, as all setups, DLL registration, and the like will be done on the virtual image.

Discussions are underway to develop an Extract, Transfer and Load (ETL) process to pull data from the MySQL database (a transactional database) and populate it to the Work In Progress (WIP) database (an analytical database). The WIP database will house data from all of our different data sources (CATI, paper, our in-house WEB self-administered data collection (EDR), and our in-house small survey data collection instruments (EDC)) and will be the one source of data for analysis and summary programs.

It is possible we will decide to turn versioning off for future surveys. There were far too many rows being written to the dataset each time a record was touched, probably because we update a user stamp in the instrument whenever anyone accesses a record. We might look into using the ETL process to create an historic database off to the side including all iterations of the data, and thus keeping only the current instance of the record in the main database. However we decide to handle it, we will want the ability to easily access both the original and current instances of the data.

We will need to make use of UDL files, due to a security feature being put in place on the MySQL databases in the not-so-distant future. A generic username and password to access the databases will be stored in a separate dataset and the password will be changed every 30 days. We're considering using one shared, temporary UDL file for each survey which will use the current password.

When more surveys are lined up for us to convert to centralized Blaise, we will start actively researching CRON jobs to automatically run certain processes at night. These could include daybatch creation, initialize, data readout (until the ETL is in place), and possibly some others.

A deployment timeline is currently in place which has the next converted survey going live in late 2010. Our other regular surveys are slated for conversion through 2012, ordered according to increasing complexity where possible.