

Manipula & ManiPlus Usage at the National Agricultural Statistics Service

David Knopf and Roger Schou, National Agricultural Statistics Service, USA

1. Introduction

The National Agricultural Statistics Service (NASS) has developed CASIC systems using Blaise III. These systems integrate data collection and interactive data editing. In addition to data collection and editing, many survey processes must be performed. Examples of these processes include file creation, data movement and report generation. Each process must be easy to execute and run efficiently. Nearly all of the survey system processes are performed by Manipula and ManiPlus. This paper will describe the many roles played by Manipula and ManiPlus, and how the setups have been written to optimize performance.

2. Uses of Manipula

Creating Data Files

One of the first survey steps is the creation of Blaise data files. Five different data files are created using four Manipula setups. One data file created in this step is the primary file used for CATI data collection. Three others are External data files read by the primary data model. ASCII data files containing sample unit information are used as input. Names, addresses, telephone numbers and other sample information is output into Blaise files.

Updating Data Files

At different points during the survey process Blaise data files are updated using Manipula. Setting the Time Zone indicator for data collection purposes is one example. Surveys that NASS conducts may include forms for sample units located across different time zones. In this Manipula setup only an Update file is defined. Address information in the form is used to compute the Time Zone indicator and to update the Time Zone field in the form. The Time Zone field is later used by the CATI Call Scheduler.

Another common updating practice is computing a value for a Secondary Key field. Other Manipula processing will then be done by the Secondary

Key. ProcessSwitch is an example of a Secondary Key field that we use. ProcessSwitch will be assigned a value based on a selected process and a subsequent Manipula setup will process by ProcessSwitch.

Moving Data

Manipula is used to move data from one data file to another data file during the survey process. Survey data resides in one of two data files. Initially all forms are in the CATI data file. During the survey, data is collected using CATI and paper questionnaires. A process we call Transition is run on a regular basis to move forms with completed data to the EDIT data file. Data from CATI moves with the form while data collected on paper questionnaires will be read into the Blaise data file from an ASCII item code data file. The code data file was prepared with a high speed, heads down data entry software. Conceptually, Transition is simple. The actual process, though, is very complicated utilizing several different Manipula setups to accomplish the operation.

Checking Forms (Integral Check)

Another function of Manipula is to perform a check on all forms in the data file or just a portion of the data file. CHECKRULES marks the forms Clean, Dirty or Suspect for the subsequent interactive edit step. Forms which are Dirty or Suspect are reviewed and modified until Clean. Usually, no further review is done for Clean forms.

Figure 1. *Manipula setup for checking rules.*

```
USES
    InstFmt      'AgSurv'

UPDATEFILE

    EDITData : InstFmt ('AgSurvE', BLAISE)

MANIPULATE

    IF (VAL (PARAMETER) = Managmnt. Batch) THEN
        EDITData. CHECKRULES
    IF EDITData. CHANGED THEN
        EDITData. WRITE
    ENDIF

    ELSEIF Managmnt. Batch > VAL (PARAMETER) THEN
        READY
    ENDIF
```

ENDMANIPULATE

A group number called the Batch Number is assigned to each form. The Batch Number is used in this process to check only a specified group of forms. Normally, checking forms is done for a group after processing through Transition. While the entire data file can be checked, the normal operating procedure is to check by group. Figure 1 contains a setup example.

Creating Reports

Manipula is also used to create several reports. These reports provide information to facilitate the survey process. During a survey period up to thirteen different reports could be generated. Typically five to seven different reports are generated, some of them several times. Input files are Blaise files and output files are ASCII. Use of the PRINT paragraph is common. Content of the reports vary.

Several reports are designed to provide information about survey progress and status at a given point in time. An example of this is the Edit Status Report in Figure 2. This report provides a count of the forms in the interactive edit data file by form status.

Figure 2. Output from Edit Status Report

```
-----  
-----  
FORMS IN THE EDIT DATASET      STATUS   OF  
02/14/1997                     DATE     :  
  
AS703                          TIME : 08:58:12  
-----  
-----  
FORMS                          FORM STATUS   NUMBER OF  
=====
```

FORMS	FORM STATUS	NUMBER OF
CLEAN	600	
SUSPECT	45	
DIRTY	155	

NOTCHECKED	200
CONVERTED TO CODE DATA	1250
NOT ACTIVE	88
=====	=====
TOTAL IN EDIT	2338

Other reports are listings that provide selected information about specific forms. A frequently used listing is the Missing Reports. This is a listing of incomplete reports and the report status. Action can be taken based on the information in this report.

Creating Code Data Files

Following interactive edit, it is necessary to convert the Blaise data to an appropriate ASCII file format for summarization. Code data is the most common format. In this process each field that must have data written to the code data file is identified by a Tag. The Tag is used by Cameleon to create part of the setup code. During this process Manipula reads the Blaise file and writes to the code data file.

3. Maniplus

ManiPlus is playing an ever increasing role in NASS survey systems. Currently ManiPlus is used for a Transactions Processor and Interactive Editing Interface. We anticipate using it to generate Manipula setups based on a specifications data model.

The Transactions Processor updates fields in the form to either withhold or reactivate a form for CATI. Fields are updated in the appointment block (CATIMana) as well a field called DayBatchSwitch. Fields in the appointment block are updated to provide CATI summary and calling history information. DayBatchSwitch is used as a Select Field in the Call Scheduler definition to exclude forms from a Day Batch. Forms are also removed or added from a current Day Batch using the DAYBATCH_DEL or DAYBATCH_ADD methods.

A second use of ManiPlus is an Interactive Editing Interface. Form level information is displayed in a list and the Data Entry Program can be executed for a form selected from the list. Identification, geographical and form status information are among the items displayed. Form status is set during the Integral Check. An editor typically reviews the list for form status Suspect and Dirty, and modifies those forms using the Data Entry

Program. The interface can display the entire data file or a portion chosen by the editor using Batch Number.

4. Improving Manipula Performance

NASS survey processing using Manipula setups with large data models, large data files and complicated survey design proved to be a time-consuming task. In the most extreme example it took nearly five hours to run one process. To make the survey applications practical to use, reducing processing time was necessary. This has been accomplished using Manipula SETTINGS and various other techniques.

Many Manipula setups don't require copying data from one data file to another or don't involve many fields of a data model. A good example of this is a report. In those setups it is common to read five to ten fields and write a similar number. In these setups CONNECT=NO is used. With this setting, only fields specified in the MANIPULATE paragraph will receive a value. Time is saved during initialization only, so this is most advantageous for large data models and small data files.

A significant amount of time has been saved processing by a Secondary Key field. Earlier we mentioned updating the ProcessSwitch field. ProcessSwitch can have the value of (1) Paper Complete, (2) CATI Complete, (3) Default, (4) Sent. When a selected process is run, ProcessSwitch will be set by a Manipula setup like ProcSw1

(see Figure 3.). A second Manipula setup will process by ProcessSwitch and then halt when forms meeting some specified criteria have been processed. Two Manipula setups are used, but the net effect is reduced processing time. This is possible because the initial Manipula setup (ProcSw1) runs extremely fast because of FILTERS and the second one then reads only a portion of the data file using STARTKEY and HALT.

Figure 3. *ProcSw1 Manipula setup*

```
USES

InstFmt 'AgSurv'

DATAMODEL MarkItFmt

FIELDS

    State :    INTEGER (2)
    ID :    INTEGER (9)
    Tract :    INTEGER (2)
    Subtract :    INTEGER (2)

ENDMODEL

INPUTFILE MarkIt : MarkItFmt ('CODEDATA. IDS', ASCII)

UPDATEFILE CATIDData : InsFmt ('AgSurvC', BLAISE)

FILTER

LFInfo

Managmnt

LINKFIELDS

    LFInfo.State =    MarkIt.State
    LFInfo.ID =    MarkIt.ID
    LFInfo.Tract =    MarkIt.Tract
    LFInfo.Subtract =    MarkIt.Subtract

UPDATEFILE EDITData : InstFmt ('AgSurvE', BLAISE°
```

```

FILTER

LFInfo
Managmnt

LINKFIELDS

    LFInfo.State = Marklt.State
    LFInfo.ID = Marklt.ID
    LFInfo.Tract = Marklt.Tract
    LFInfo.Subtract = Marklt.Subtract

MANIPULATE

IF RESULTOK (CATIData) THEN
CATIData.Managmnt.ProcessSwitch := PAPICmpl
WRITE (CATIData)
ELSEIF RESULTOK (EDITData) THEN
EDITData.Managmnt.ProcessSwitch := PAPICmpl
WRITE (EDITData)
ENDIF

ENDMANIPULATE

```

The FILTER section is used to read only a portion of a data record, rather than the entire record. Filtering can be done for an individual field or for an entire block.

In the FILTER section one or more fields can be specified. Only the specified fields are available in the setup. Increased processing speed can be realized for large data models and large data files. In Figure 3, two blocks are read from the input file. These two blocks account for less than 5 percent of the data record.

STARTKEY is another setting used extensively to save processing time. This is used when only a portion of the data file will be processed. STARTKEY is used with the setting KEY. The processing order of forms is determined by KEY. Processing begins with the first form with a KEY value equal to or greater than the STARTKEY values. When all forms that match the STARTKEY value have been processed the setup is conditioned to HALT. This is most beneficial when a data file with a large number of forms is being read. Applying this technique at NASS has reduced the processing time for some steps 50-75 percent. Figure 4 shows the use of KEY and STARTKEY. A typical setup like this one will read 100-400 forms of a 2,500 form file.

Figure 4. Manipula setup showing STARTKEY

```

SETTINGS
AUTOREAD = NO

USES
InstFmt 'AgSurv'

UPDATEFILE EDITData : InstFmt ('AgSurvE', BLAISE)

SETTINGS
KEY = SECONDARY (BatchNumber)
STARTKEY = (100)

OUTPUTFILE BATCHData = EDITData
('BATCH\AgSurvE', BLAISE)

AUXFIELDS (GLOBAL)
SomeSent : INTEGER (1)

PROLOGUE
EDITData. GET (VAL (PARAMETER))

```



```
MANIPULATE
REPEAT
IF (EDITData. RESULTOK) AND
(EDITData.Managmnt.Batch = VAL (PARAMETER) THEN
... other instructions“
ELSE
HALT
ENDIF
UNTIL EDITData.EOF
ENDMANIPULATE
```

A number of processes use ASCII files as input. Efficiency can be improved when processing an ASCII file against a Blaise file by defining the ASCII file as the first input file as shown in Figure 5. In this way, only the forms in the Blaise file that match the ASCII file are read. In the Figure 5 example, the ASCII file will typically have 100-400 records and the Blaise file will have 1000-3500 forms. This eliminates reading several hundred forms.

Figure 5 : *Defining ASCII file before Blaise file*

```
INPUTFILE MarkIt : MarkItFmt ('CODEDATA.IDS', ASCII)
```

```
UPDATEFILE TrackOut : TrackFmt ('Tracking', BLAISE)
```

5. Conclusion

NASS has found Manipula and ManiPlus to be valuable tools for its CASIC survey systems. Using features to improve processing speed, NASS has been able to process large volumes of data very efficiently.