Using metadata in Manipula and Maniplus

G.J. Boris Allan, Richard Frey, Jim O'Reilly

Westat

14th International Blaise User's Conference

Introduction

- Traditionally, Manipula/Maniplus has been used to access data and Cameleon for metadata
- Cameleon produces text files as output and is used to generate Manipula programs of the extracted metadata to export a database structure and related information
- Beginning with Blaise 4.x, powerful metadata access functions were added to Manipula, making Cameleon redundant for many important Blaise processes.
- We will discuss three recent applications of this technology.

Removing remarks in rollover programs

- Longitudinal surveys often require data from one or more previous rounds be carried over to current round
  - Remarks are also copied into the current round, making it difficult to distinguish current remarks from previous round remarks
  - A solution is to set remarks for all fields in the current database to empty as part of the data assignment process for the new round
  - A Manipula procedure clears out the prior round remarks (called in the MANIPULATE section by RemoveRemarksBlockProc(''))

Removing remarks

```
PROCEDURE RemoveRemarksBlockProc
  PARAMETERS
    BlockName : STRING

  AUXFIELDS
    FieldName : STRING
    SectionName : STRING

  INSTRUCTIONS
    FieldName := Present.GETFIRSTFIELDNAME(BlockName)
    REPEAT
      IF (Present.GETFIELDINFO(FieldName,
     'BASEFIELDTYPENAME') = 'BLOCK') THEN
        RemoveRemarksBlockProc(FieldName)
      ELSE
        Present.PUTREMARK(FieldName,'')
      ENDIF
      FieldName := Present.GETNEXTFIELDNAME(FieldName)
    UNTIL (FieldName = '')
ENDPROCEDURE
```

- Identifies first field name for named block (initial call specifies an empty block, taken to be top level)
- Repeats looping until no next field name (end of block)
- Calls procedure with new block name, if field is a block
- Makes remark empty if field is not a block
- Gets next field name

Finding where instruments stop

- Studies often define a partially completed case status when the interview is mostly completed, but not totally.
- Various processes need the last question in the interview
- A Manipula program identifies this information for all partially completed cases in a study
  - Partially completed given operational status between 80 and 89.

Finding where instruments stop (2)

```
SETDESCRIPTION(ParentID)
  IF (SUBSTRING(CAPIStatus,1,1) = '8') THEN
    InputFile1.CHECKRULES
    FieldName := InputFile1.GETNEXTROUTEFIELDNAME('', 'ASK')
    REPEAT
      OutputFile1.EndFieldName := FieldName
      FieldName := InputFile1.GETNEXTROUTEFIELDNAME(FieldName, 'ASK')
    UNTIL ((FieldNAME = '') OR  ((InputFile1.GETVALUE(FieldName) = '' ) AND
     (InputFile1.GETVALUE(InputFile1.GETNEXTROUTEFIELDNAME(FieldNam
     e, 'ASK')) = '' )))
    OutputFile1.WRITE
  ENDIF
```

Example of output

XXXXXXXX|CFQ.CFQ310_F
XXXXXXXX|Fsq.OriginsKeyParent[2].WhereBorn
XXXXXXXX|PPQ.PPQ270_F
XXXXXXXX|HEQ.HEQ400_F
XXXXXXXX|Fsq.RelTable.Relations[3].RelationToChild
XXXXXXXX|Fsq.OriginsKeyParent[2].HowOld
XXXXXXXX|NRQ.Questions[2].NRQ122_F
XXXXXXXX|Fsq.OriginsKeyParent[2].WhereBorn
XXXXXXXX|SSQ.SSQ010x_F
XXXXXXXX|Fsq.EducationsKeyParent[2].FSQ221_F
XXXXXXXX|HEQ.HEQ370_F

- Select only partial completes based on operational definition
- Check the rules for that case
- Get name of next field on route that is ASK field, starting from beginning.
- Repeat loop for next section

.

- Store current field name in output file buffer (EndFieldName)
- Get next ASK field on route
- Repeat code loop unless no more fields on route with data
- Write contents output file buffer.

WesBlaiseSAS

Auto conversion of Blaise data to SAS

- Many studies need to convert a Blaise database with its datamodel into a SAS dataset with variable descriptions, formats, and with a matching text input file

- Scale of conversion can be large
  - —One study—146,000 data fields, labels etc.

- WesBlaisetoSAS generalizes the process

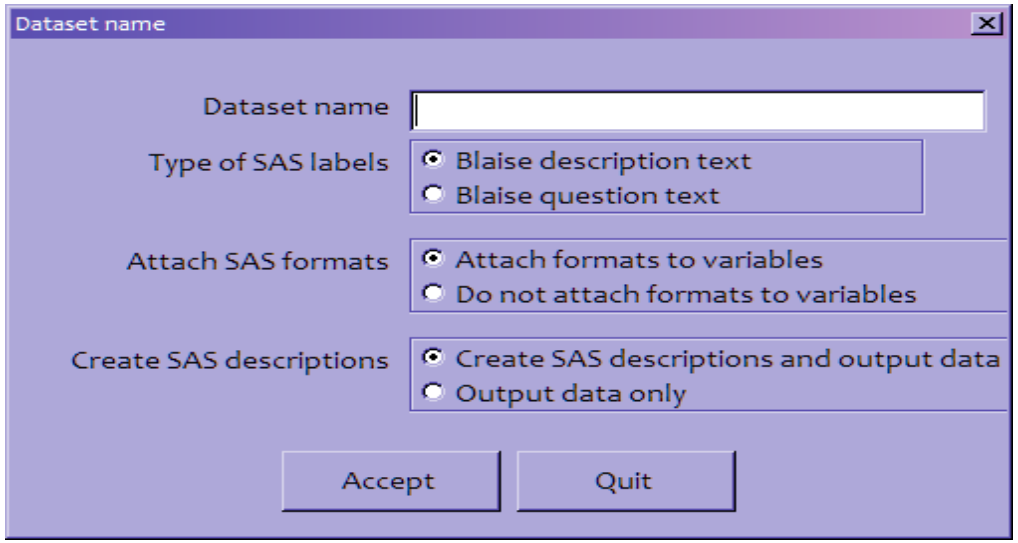
- Uses two Maniplus programs
  #1 Asks for file names and folder locations, and
  #2 Uses a variable datamodel and many temporary files to generate the SAS data definitions

WesBlaiseSAS (2)

- InstallWesBlaiseSAS.exe installs components in C:\WesBlaiseSas
  - Manipula.exe (used by msu files to read/write Blaise information)
  - README.TXT
  - RunSAS32.msu (collect details about files and folder locations)
  - SAS32BlaiseDriver.msu (uses Blaise datamodel descriptions and Blaise data to produce a SAS program, which opens in SAS)
  - WesBlaiseSAS.lnk (executes RunSAS32.msu to collect file and folder information and produce the SAS program)

WesBlaiseSAS processing

- Select Blaise datamodel, database, folder for SAS program
- Choose options and provide a dataset name



- Maniplus process CreateSASInstructionsFromBlaise shown on page 4 and 5

WesBlaiseSAS (3)

- SASDataDescriptions uses
  - Blaise metadata to produce SAS descriptions of data
  - Blaise data to produce a text file that matches the descriptions already produced (pp 5-6)
- On completion, SAS screen displays the appropriate output.
- Using the Blaise NCS01 example datamodel the process delivered
  - NCS01.ASC – the ascii data file
  - NCS01.SAS – the SAS code

WesBlaiseSAS (4)

```
PROC FORMAT;
  VALUE W000001W
    1 = "Original data entry from paper"
    2 = "Verification of paper data entry“
…

LIBNAME SAVEFILE
    "C:\WesBlaiseSAS_Test\";
DATA SAVEFILE.ncs01;

INFILE "C:\WesBlaiseSAS_Test\ncs01.ASC"
    LRECL = 332;

INPUT
  Batch_1       1-3
  ModeProc_1       4-5
  DataSource_1       6-6
…
```

```
LABEL
  Batch_1 = "Batch"
  ModeProc_1 = "ModeProc"
  DataSource_1 = "DataSource"
  FormStat_1 = "FormStat"
  TimeStart_1 = "TimeStart“
…

FORMAT
  ModeProc_1    W000001W.
  DataSource_1    W000002W.
  FormStat_1    W000003W.
  TimeStart_1    W000004W.
…
```

Cameleon – outmoded or still useful?

- Metadata manipulation part of Blaise from the beginning
- In Blaise 2 'the setup generator‘ was implemented by Adriaan Hoogendoorn and colleagues
- Cameleon followed
- Cameleon still valuable for stand-alone situations
  - Language a bit strange at first, but powerful
- Manipula/Maniplus for metadata can be clunky in part
  - But integrates smoothly with external systems

Questions?