Part 9 - Data Processing and Data Delivery

Laura D. Yoder - Survey Research Center, University of Michigan

Over the course of the transition of the HRS from Blaise 4.8 to Blaise 5, we have investigated many ways to access data and adapt our old processes. In this paper, we will discuss the ways we modified well-defined data processing procedures, and the challenges and successes we had along the way.

1. Survey Preload

With an established panel sample, we often know some information about the people we are collecting data from and we look to use this information throughout an instrument. This includes but is not limited to name, gender, family composition, and previous employment. In previous iterations of the HRS, these preload data were loaded into the sample management system as a caret delimited string and pushed into the Blaise 4.8 instrument when a survey was launched by an interviewer. Since we are now working in two different environments, one offline and one online, we have to maintain preload in two separate ways.

1.1 Offline Survey Preload

Similar to what we did in the past, preload for the offline sample is still handled by the sample management system. Since this sample is still initiated by an interviewer, no changes were needed to the process. The changes that were made were to the scripts that run once an interviewer launches an instrument. Manipula scripts and custom sample management programming are used to push a caret delimited string from fields in the sample management system into the individual Blaise 5 instrument.

1.2 Online Survey Preload

For the online sample, the preload has to be loaded directly to the server version of the instrument. Because web respondents bypass the sample management system, we could not use the same process for preloading the instrument as we do for offline sample. In order to preload the server instrument, we have instituted the idea of a “preload data model.” This unique compiled data model contains a subset of the blocks in the main instrument that are required to map preloaded information back to the main database. For the HRS study, the creation of the separate data model (and associated complex Manipula script) falls to the Blaise instrument programmer and the data manager group uses the script to import the preload information. Below are example pieces of this script.

```plaintext
SETUP HRS2018_ServerPreload
SETTINGS
  DESCRIPTION  = "Manipula Setup - Import HRS 2018 Preload into Web Version"
USES
  HRS18 'HRS18.bmix'
```
DATAMODEL InHrs2018Data

FIELDPROPERTIES
    Remark: Open
    IsVisited: TIsVisitedFieldProperty
    AlienActionEvent : string

ATTRIBUTES = DONTKNOW, REFUSAL

INCLUDE "HRS18SpecialAnswers.incx"
INCLUDE "HRS18_Type.incx"
INCLUDE "HRS18_SCV.incx"
INCLUDE "HRS18_Basis_Tables.incx"

FIELDS
    SampID /"SAMPLE ID" : STRING[10]
    HHID /"HOUSEHOLD ID" : STRING[10]

{tShared}
    Preload_RTab : ARRAY [1..2] OF B_RTab
    Preload_HH : B_HOUSEHOLD
    Preload_Respondents : ARRAY [1..3] OF B_People
    Preload_Children : ARRAY [1..50] OF B_People
    Preload_HHMembers : ARRAY [1..20] OF B_People

{tCAPI}
    Preload_SCV : B_SCV
    Preload_RVARS : B_RVARS
    Preload_PastPens : ARRAY [1..10] OF B_PastPens
    Preload_Job : ARRAY [1..10] OF B_Job
    Preload_Hlth_Plan : ARRAY [1..3] OF B_HlthPlan
    Preload_RSiblings : ARRAY [1..20] OF B_Siblings

AUXFIELDS
    FLJ535, FLJ005 : STRING

ENDMODEL {InHrs2018Data}

INPUTFILE
    MyInputFile: InHrs2018Data ("preload.asc", ASCII)
    SETTINGS
        SEPARATOR = "^"

OUTPUTFILE
    HRS18Output: HRS18 ("HRS18.bdbx", BLAISE)
    SETTINGS
        MAKENEWFILE = NO

MANIPULATE
    HRS18Output.WRITE

ENDSETUP //HRS2018_ServerPreload
1.3 Considerations Regarding Survey Preload
While both of the above processes work, there is a lot of preparation involved to prepare two different sets of preload and to keep them up to date. First, the Manipula scripts need to be recompiled each time there is a new data model, even if the preload has not changed. Second, updating preload requires a change to the caret delimited string (offline) and the creation of a new Manipula script that references only the fields that need updating (online). Both of these processes are prone to error and require substantial testing any time a change is made.

As we move toward working in one sample management system that handles self-administered surveys (online), pooled CATI surveys (online), and distributed CAPI and/or CATI surveys (offline), the preload process will likely be reduced to only having to load and maintain the server instrument.

2. Survey Migration

At certain times during the data collection period, a new version of the instrument may need to be released. This could be due to an error in the original instrument, the need for an additional field, or many other reasons. The HRS calls this a survey migration or update.

To implement the change, data must be migrated from the old version of the instrument to the new version. If a respondent has already started the survey in the old version, we need to ensure that their data are retained and that they are able to resume the survey from the same question where they originally suspended. Like with preload, this process is different depending on whether we are dealing with the offline or online instrument.

2.1 Offline Survey Migration
Whether offline or online, a Manipula script is needed to complete the migration process. For offline, a Blaise to Blaise script is sent to the laptops, and runs the next time the instrument is launched by the interviewer. In the case of multiple migrations, it is important that each laptop have a Manipula script that migrates from each older version to the newest version. This is needed in case for instance, a sample line is started in version 2 and is not touched again until version 5. The migration process then needs to update that instrument from version 2 to 3, 3 to 4, and finally 4 to 5 before the interview can be resumed.

2.2 Online Survey Migration
For the online instrument, the same Blaise to Blaise script that is used offline can be used during the migration process. However, instead of sending the script down to the laptop and calling it upon launch, we choose a time to migrate the data on the server to the newest version. We deactivate the instrument on the server, allowing anyone who is currently taking the survey to complete but not allowing any new respondents to start, and then run the migration Manipula script. We do not need to maintain multiple versions of the script on the server side because all of the data are upgraded with each migration, regardless of whether a respondent has started or not.

2.3 Considerations Regarding Survey Migration
There are some additional considerations to be taken into account when doing survey migration. Most importantly is in regards to the session database. In Blaise 4.8, we only had to complete the above steps to do a survey migration, but the introduction of the session database in Blaise 5 caused us to rethink our process. Since it is import that we allow our respondents to resume the survey from where they suspended, we had to do two things:

1. Program the instrument to save data to the main database upon survey suspend as well as survey completion.
2. Delete the session database after migration so that upon resume, the session data are recreated from the main database.

Additionally, custom programming logic was added to the instrument to ensure that partial respondents did not have to have to resume the survey from the beginning and click through to their last visited field. These steps have allowed us to preserve data for respondents who have started the instrument, and to eliminate their burden of having to start over.

3. **Survey Merge**

Because of some design decisions made for the HRS, as well as the design of Blaise 5, we must merge the survey data into a single database before we analyze the data. This is similar to our process using Blaise 4.8. To accomplish this goal, we use a combination of a custom application and Manipula scripts.

3.1 **Offline Survey Merge**

Survey data in the offline environment are stored as individual .dbbx and audit data are stored in individual SQLite databases. In order to combine data from a single interview with other interview data, the Interview Data Merge application is used. Each instrument has its own set of criteria for when to merge a case and where to put the final data. A merge Manipula script is run to combine the individual .dbbx with the master data, and the SQLite data are converted to SQL Server data, by the Interview Data Merge application, and stored in a single SQL database on the Blaise server. Below is an example of this script.
3.2 Online Survey Merge
Since the online instrument is stored as one database on the server, there is no need to merge the data (i.e. it is already merged).

3.3 Considerations Regarding Survey Merge
Once the merge process is set up, it runs via a nightly batch process. While this requires very little overhead in the long term, it is important to note that there are many steps to set up the merge correctly. This includes having the appropriate ODBC connections to SQL Server databases, having a SQLite database reader installed, creating and compiling a Manipula script for each instrument and each migration, and making sure all the correct data locations are in place.

In the future, the survey merge will no longer be necessary as the main instrument will be on the server and the upload/download and sync processes will allow for all instrument data to be stored in one master .bdbx.

4. Main Data Delivery
Per request of our client, data are delivered as a single .bdbx of completed cases only. We also deliver remarks in a separate excel file. This happens every week at the start of the project and every two weeks once the project is more stable. In order to accomplish this delivery, multiple steps and Manipula scripts are required.

4.1 Offline Data Delivery Preparation
Due to the use of the Interview Data Merge application discussed above, little is needed to prepare the offline data for delivery to the client. The master .bdbx already contains only completed cases (criteria defined in the merge application) and has already been migrated to the most recent data model.
4.2 Online Data Delivery Preparation
The online data take a little more manipulation than offline for data delivery. Since all data (complete and incomplete) are stored in the server .dbx, we must first exclude incomplete cases. To do this, we run two Manipula scripts, SIDOut and DeleteCases.

The SIDOut script reads the database and exports a text file list of all cases that are not complete.

```
DATAMODEL Subsetfields
  ATTRIBUTES = DONTKNOW, REFUSAL
  Type
    tComplete = (Done(1),NotDone(2))

FIELDS
  SampId : STRING[10]
  Complete : tComplete

ENDMODEL

UPDATEFILE IwData : MiniModel1 (HRS18.dbx, BLAISE)
SETTINGS
  AUTOCOPY = No

OUTPUTFILE subset : Subsetfields(SIDOUT.txt, ASCII)
SETTINGS
  SEPARATOR = '^'
  HEADERLINE = YES

MANIPULATE
  IwData.READNEXT
  WHILE IwData.RESULTOK DO
    IwData.COPY
    Subset.Complete := IwData.Complete
    IF IwData.SampID = 'XXXX'
      OR IwData.SampID = 'ZZZZ'
      OR IwData.Complete <> 1 THEN
      Subset.WRITE
    ENDIF
  IwData.READNEXT
ENDWHILE
IwData.Close
```

The DeleteCases script then reads that list and deletes any cases on the list that it finds in the main database (see below).

```
USES
  HRS18

DATAMODEL Small
  FIELDS
```

239
SampID: STRING[10]
ENDMODEL

UPDATEFILE
BigFile: HRS18 ('HRS18', BLAISE)

INPUTFILE LookUpFile: Small('SIDOUT.txt', ASCII)
LINKFIELDS
SampID

MANIPULATE
IF LookUpFile.SEARCH(BigFile.SampId) THEN
    BigFile.DELETE
ENDIF

All of these steps are of course performed off of the Blaise server so as to not interfere with ongoing data collection. The resulting .bdbx is then ready to combine with the offline data.

4.3 Combined Data Delivery and Considerations Regarding Data Delivery
To combine the offline and online data, we once again employ the Manipula script that runs as part of the interview data merge process. The script allows us to merge the online cases into the offline .bdbx to create one large database. A caveat is that both offline and online versions need to be on the same data model. If that is not the case (rare), the version that is older must be migrated to the newer version before the merge takes place.

Since our client requests the full .bdbx as output, in theory there is no need to go any further. However, we must also provide remarks and other field properties as separate excel files. To do this, we proceed with a basic data out Manipula script. All of the data are exported in wide format in a .txt file and a separate .fps file is produced. Finally, we read the .fps file into SAS and filter the file so that only the properties we want (Remarks and IsVisited) are included in our resulting excel files.

Due to a bug in previous versions of Manipula, this data delivery must completed using version 5.4 even though our instrument is using 5.3.1501. Also, once both offline and online data are stored in the server .bdbx, we will only need to delete incomplete cases and process the field properties for data delivery.

5. Paradata Delivery

The audit data provided by Blaise are a useful source of data and provide valuable insight into the actions our interviewers and respondents take. Once processed, the audit data can be used in many ways including, but not limited to, troubleshooting/QC, instrument timings, and reports. For the HRS, we provide the raw audit data in SQL Server databases as well as parsed data that have been combined with data from other sources.
We parse the audit data using a series of SQL stored procedures and then store the data in SQL Server databases for use by other applications. For more information about the process and output, see the paper Transforming Survey Paradata (Piskorowski, Simonson, Yoder, IBUC 2018).

The parsed audit data are then used to create timings reports for the client. We use SAS to aggregate and analyze the parsed data and to produce excel files that can be used for additional analysis or decision making.

6. Summary

In the end, while transition of the HRS from Blaise 4.8 to Blaise 5 is still in progress, we have found many ways to adapt existing processes to the new world. It has been frustrating at times, often because of both our own decisions and the limitations of working with a new product, but we hope that the processes we put in place for the HRS can be used across other projects.