Legacy Michigan CATI Sample Management System – Mixed Mode CATI/Web

David Dybicki and Peter Sparks, University of Michigan

This paper examines three distinct areas for implementing a mixed-mode survey: SMS, Blaise IS, and routines needed for mixed mode. The project used for implementing the mixed mode has been referred to generically.

1. SMS

SMS is short for Sample Management System, and was developed by a small team at the University of Michigan over a short period of time and went into full use somewhere in the fall of 2001.

1.1 History

SMS was designed using paper prototyping. Paper prototyping was not just a buzz word but proved useful for the design process. It gave the end user control over the final outcome of the system, in terms of look and feel and actions. This dramatically sped up the process of development.

The first project was deployed in parallel using the old paper handling method, and the "new" case delivery system. The project had monthly data collection requirements and proved optimal for comparing results.

Since the system is programmed in Maniplus, it made changes simple and the code was portable, maintainable, and could be tailored to most project needs. SMS has the ability to work with any file type that is supported by BOI files, and has worked with SQL server as well as native BDBs.

The design separated the SMS-related fields ("INHERIT CATI") from the survey-specific datamodel. This design allowed for complete sharing of the survey datamodel between modes (CATI & SMS/CAPI & SurveyTrak[1]).

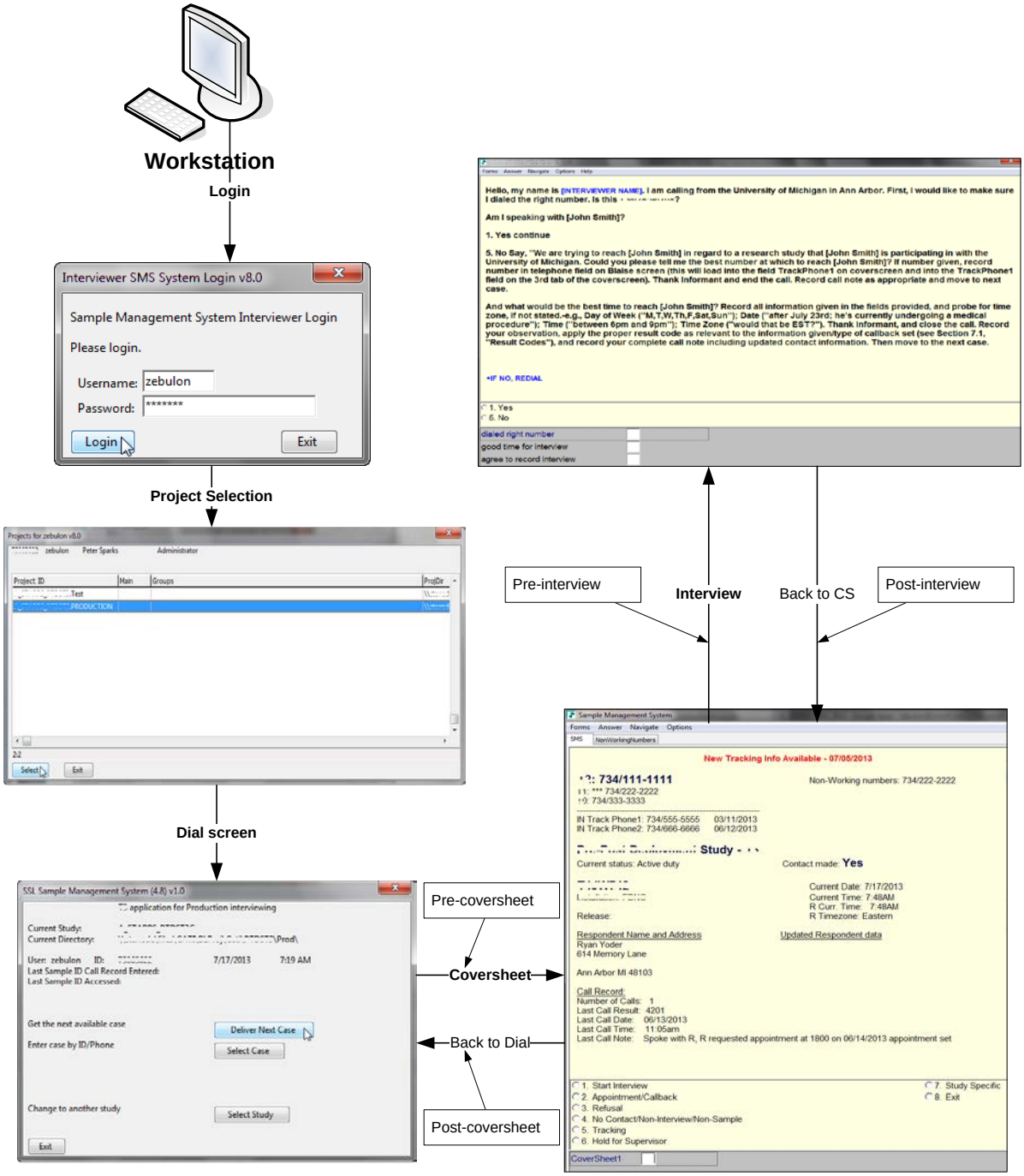
1.2 Implementation

SMS was developed around the Blaise call scheduler "out of the box" using Maniplus. Three major scripts are run: SMSPicker, SMSCati and SMSSuper.

1.2.1 Interviewing process

The diagram below shows the interviewing process: logging in via SMSPicker and selecting a project, then interviewing using SMSCati. SMSCati runs the call scheduler to retrieve the next available case according to the parameters specified in the CATI specifications. A very minimal interface allows the interviewer to receive the next available case via the call scheduler, to select a case via phone/primary identifier, or to select a project. The coverscreen replaces the "make dial" screen.

CATI Interviewer Process



An important design feature is the ability to run additional Manipula/Maniplus scripts between SMS, the coverscreen datamodel[2] and the study datamodel[3]. This is key to implementing a mixed mode survey between CATI and web. This implementation of SMS and mixed mode uses the post-coversheet (data is written to the web management database) and the pre-coversheet and post-interview (sets section flags in the CATI and web interviews, respectively).

1.2.2 Management mode

The supervisor mode, SMSSuper, works with sample (release, review, edit, assign), with users (groups, logins, projects), with call records and result codes[4], call history, readback & interview mode. In the interview mode the mixed mode flags (see 3.9.2 below) are also checked and set.

1.2.3 System audit trail

Every significant action of the supervisor or interviewer, such as logging in to the system, retrieving a case, changing a user's role, etc., is recorded in trace files. This helps resolve the problem of incorrect/missing data in the case management.

1.2.4 CATI audit trails

Standard ADT files are stored for each interview. They are used for interview timings, and to possibly recover an interview in case that case becomes corrupt as well as to debug complex logic flows. It can be used to help determine interviewer behavior for problem screens.

1.2.5 CARI recordings

The recording of audio has been set to infinite to capture all the dialog between the respondent and the interviewer for quality control. Screen captures of the question are also stored. Sometimes, the dialog cuts out abruptly when the interviewer presses Enter too quickly and leaves the question. We have found the CARI recordings to be very useful.

1.3 Coverscreen uniformity

Though it would seem very restrictive to maintain only one structure for all SMS projects, each coverscreen is actually very flexible. All projects maintain the same base coverscreen structure. This is a datamodel that contains the INHERIT CATI command, as well as standard fields for use within the system. Study-specific questions are added using auxfields, and any study-specific data, such as name, phone number, address, and age, are stored in additional arrays: UserValue[], KeyStats[], IwerObs[], ContactObs[].

This coverscreen also contains result codes for the interview, such as a "Ring, no answer" or an initial refusal. This means that any reports, utilities, or routines created to work for one study will work for all studies in SMS.

1.4 Other functions

Extensions to SMS include programs that create reports for managing the sample, track interviewer progress and hours-per-interview. The task scheduler is used for creating new day batches automatically, running Hospital to rebuild databases to keep them healthy, and executing Manipula scripts that are specific to the project, such as setting sub-priorities[5], moving sample between modes, and so forth. There are built-in filters in SMSSuper that allow supervisors to quickly locate cases and work with them.

1.5 New data model releases

Data model changes solely concern the study data model and not the coverscreen. As previously mentioned, the coverscreen data structure is the same between all SMS projects. Most migrations are standard: backup the data, use a Blaise to Blaise Manipula script (via the wizard), and then replace the production data model and data with the migrated data. This same script is also used for SurveyTrak field projects. Project specific Manipula

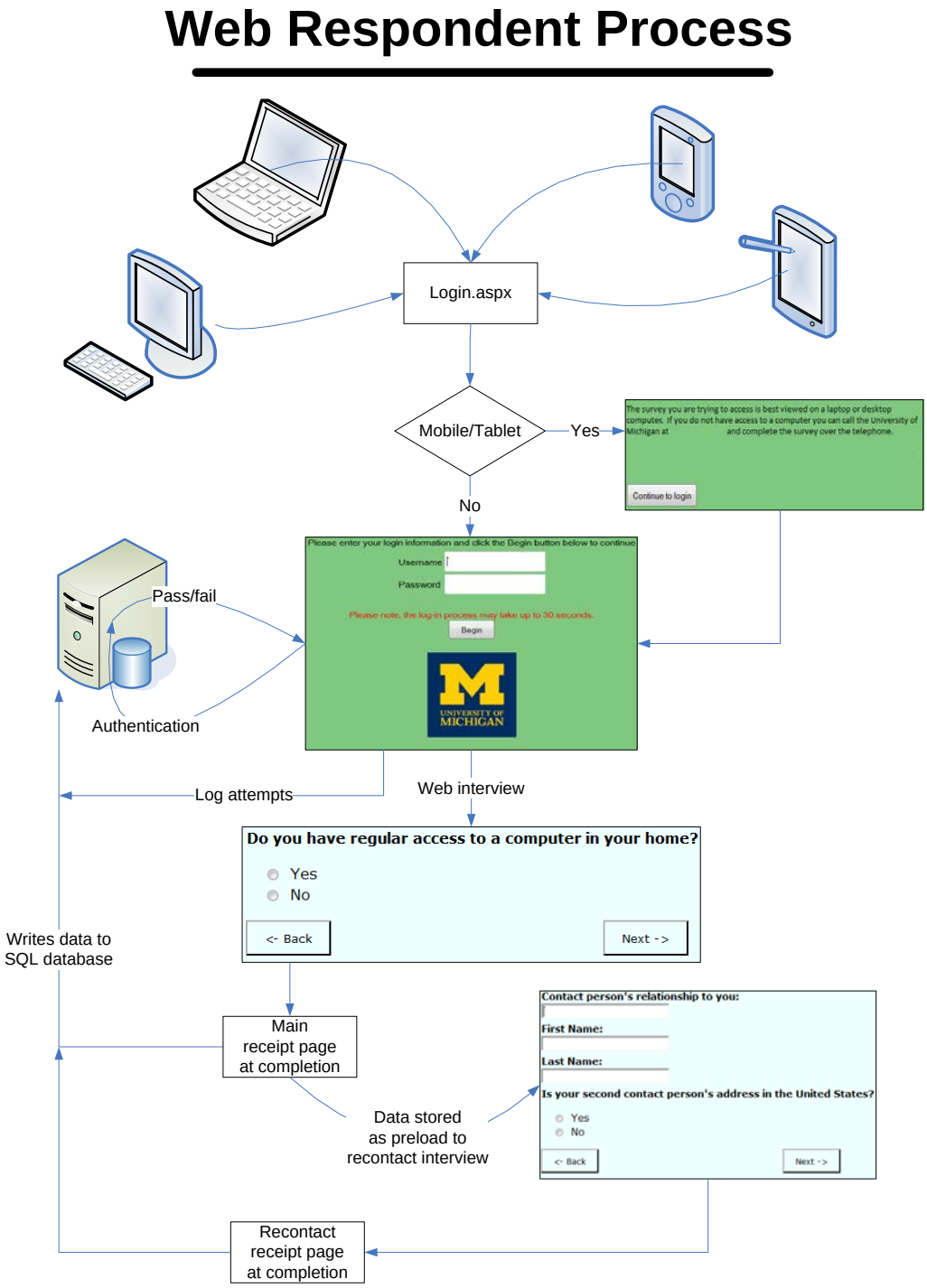
scripts, such as the pre-coversheet, post-coversheet, and post-interview routines are also have to be prepared again.

1.6 Legacy limitations

Like all legacy systems, there are a number of items in SMS that are could be improved. The interface is functional but looks and feels old. An out-of-date VB6 DLL is used by the system to manage a shared text file to achieve fast reads in a few files. However, this should be changed by storing the data in a relational database. Project maintenance can be time-consuming because of constant monitoring to ensure daily/nightly batch files run correctly. Reports are functional but need to be updated or replaced. There are many files and scripts, and although it provides flexibility it is also more difficult to maintain and catch errors. And, because of this, SMS is fragile and one failure, such as a locked or missing file, can bring a project to a halt.

2. Web

The process for the respondent using entering a Blaise IS interview is illustrated below.



2.1 Survey process

The respondent enters the survey at the login page via some device (i.e. PC, laptop, cell phone). The login is a .Net web application, and it first checks the user's device. If it is a mobile device then a

warning page is shown stating the survey has been formatted for desktop browsers. The login page queries a backend SQL database with username & password, and if all is well the database returns the ID of the interview, different than the username, that is used to start the Blaise IS interview.

Note: if the interview has been marked as complete (a status from either SMS or web), or the case has been locked in web, then the web management system sends an invalid ID back to the login page.

2.2 Paradata

Similar to the function of ADTs, the web mode uses client side paradata (journaling) to capture information about the user's actions on a web page when answering. This data is stored in a database on the data server for timing analysis. It also can be used to help recover the data within an interview. It can be used to help determine respondent behavior for problem pages.

The recontact interview is not journaled. The recontact interview contains sensitive information, such as respondent contact information and payment. The purpose of separating the main interview and the recontact interview is to maintain a separation of respondent identifying information from survey data.

2.3 Web management database

The recontact interview is always conducted at the end of the main interview. At the end of each of the interviews, the Blaise IS receipt page is run. The pages have been modified to not display any information, but rather write key variables and other data back to the web management database.

Note: none of the flag settings occur at this point; all of those actions are taken with SMS.

2.4 Main and Recontact

2.5 New datamodel releases

A more complex process than migrating SMS is used. The web mode is made inactive via the Internet Server Manager, the web surveys and data are backed up, the new survey is deployed, the data is migrated back to the production area, and the web mode is again made active. The downtime for the web survey is planned and short, and occurs during pre-determined light usage times.

3. Mixed Mode

We use the definition of a survey that has been implemented for CATI (interviewer-assisted, Blaise 4.8, using the DEP.exe) and web (respondent self interviewing, Blaise IS 4.8, browser). The survey is automatically or programmatically sharing sample between the two modes and the case can be switched between modes repeatedly.

3.1 Blaise IS management

Key to managing the sample in web is the management database. It is used for storing key data values, such as respondent name, address, phone, CATI/web mode, contact information, and key variables from the survey. The management system also generates email invitations, reminders, text messaging, and form letters. Respondent incentive payments are also handled through this system. Data from SMS is pulled into the system every five minutes. Data from the web interview is written to the database at the end of each interview's completion.

3.2 Case locking

After careful deliberation, it was decided to handle case locking only within the CATI mode; all cases loaded in the web are always available. The sample is duplicated between CATI and web, and in CATI the sample is released by replicate for data collection. This means that a respondent could theoretically take a web survey at the same time as a CATI survey, but it is unlikely.
This design allows respondents, who are initially assigned a web interview, to call a toll-free number and to request an interviewer-assisted interview.

3.3 Implementation of the survey in CATI and web

The question order and code frames were kept as similar as possible in order to maintain consistency, even though there is not a one-to-one migration of data between the modes. CATI mode has fewer questions than web, and the types for some questions are also different but the field names remain the same.

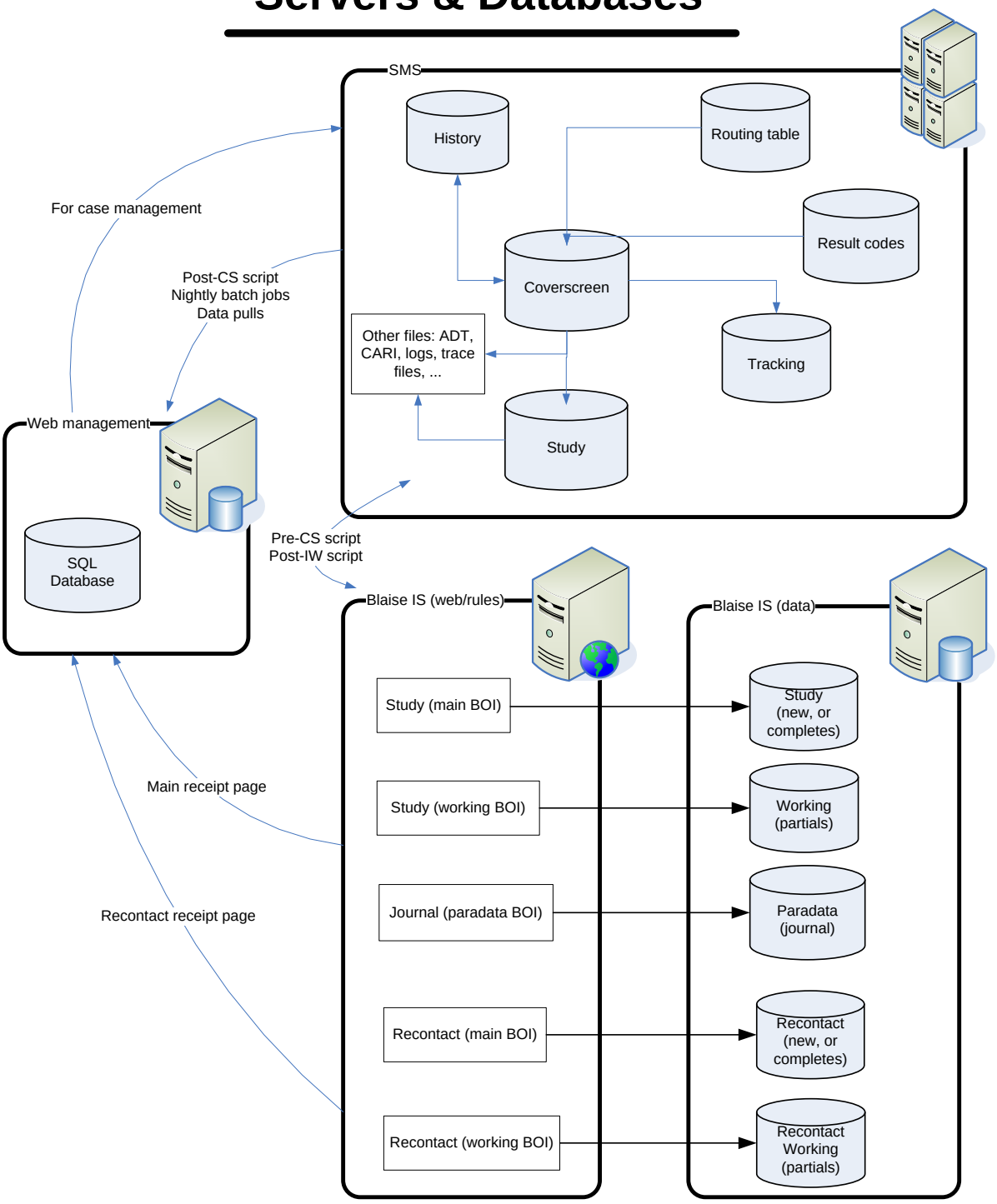
3.4 Grouping and Formatting

While web mode allows for many questions on the same page using grids, CATI standards are one question per screen. Hence, lead-in questions, optional text to read, interviewer instructions, question help, DK/RF, and commentshave a very different look and feel between the modes. The table below compares some of these differences.

Criteria	SMS (CATI)	Blaise IS (web)
History search	Shows call history for the case	None
Help	Context sensitive	None
DK/RF	Enabled	Disabled
Comments	Enabled	Disabled
Navigation	Page up and down, arrow keys, home, end, enter key	Page up and down, arrow keys, space bar to select, Prev and Next buttons.
Questions per page	One question per page, entry in form pane.	Multiple questions per page, entry on page.
Series of questions	Lead in questions parenthesized in followups	Questions series presented in grids
Mode	Iwer assisted interview	Respondent self-interview
Required	All questions are required	Respondent may skip any question

3.5 Data storage

Both CATI and web use BDBs to store the data collected. However, there are many differences between the CATI and web. SMS stores partial interviews in the same study database, while the web stores it in two data bases (main and working). Additionally, SMS also has history, tracking, a routing table (group/individual routing) and result code databases. Blaise IS stores respondent actions and pages visited in the paradata database (journaling), while SMS stores this information in ADT and CARI files. In addition, relative BOI files are created by Blaise IS when deploying the survey package to the production web server, which in turn point to the production data server.



3.6 Testing methods

Testing the mixed-mode system involved typical testing within the SMS and Blaise IS, as well as testing all of the interactions in the mixed mode. Scenarios were provided to guide testing. Both modes used a question in the survey to select which section(s) to test, including user-edited preload.

The following programs and utilities were used.

3.6.1 CAI Testing Tools (CTT)

CTT, a utility developed by the University of Michigan, provides an interface for group testing of a survey in both CATI and Blaise IS. It creates a local copy of the Blaise DEP and the survey, and stores tester's comments and cases on a networked data base. CTT also contains reports of outstanding items per user and project, priorities, data model date/time, screen shots, etc. The testing through CTT was not routed through the .Net login page.

3.6.2 Email invitation

An email invitation is s\Sent out by the web management system and contains a hyperlink to the survey. The tester then clicks on the link to be taken to the login page for entry into the survey, just as the respondent would do.

Initially, test lines were loaded on the production server, and then cleared before production began. However, this meant once production started the test environment was no longer available. Later, a parallel project was added that duplicated the production survey in a safe manner.

3.6.3 Blaise Internet Server Manager

Programmers tended to use this method to check the survey before releasing it for testing via CTT. This did not use the.Net login page.

3.6.4 Control Centre

This was used by the programmer to test the CATI survey.

3.6.5 Survey Preview

For the Blaise IS survey used by the programmer, the preview mode in the Internet Specifications is a useful way to browse through all the pages of a web survey and look for obvious errors in pagination, layout, text enhancements, tables, and so forth. However, fills are not shown because there is no data associated with the case.

3.6.6 Modelib preview

This provides another way to look at all the pages of a CATI survey, with the same limitation for fills as the Blaise IS survey preview.

3.7 Security

Interviewers in SMS must log in to the server in order to work. They have limited access to folders within the server, related only to the surveys they are assigned. They then log in to SMS – utilizing their username/password for SMS (not the same as the server login). Only after this process can they start the interview.

For the web, the respondent must use the provided username/password. The authentication page queries a back-end database with this information and redirects the browser to the actual Blaise IS survey with the returned primary key. Username, password and primary key are all random character/digit combinations to minimize an attack by generating username/passwords.

3.8 Servers

There are a number of servers involved in the mixed mode design. Each has its own important role.

3.8.1 CATI

A separate production server is used that contains SMS, datamodels, and databases. The CATI data is stored separately than the web databases.

3.8.2 Web

Two servers are involved: web/rules server and a data server. The web server contains no data, but contains relative BOI files that are created by the Blaise Internet Server Manager that point to the production data server. Both servers also have the datamodels (study, working, paradata).

3.8.3 Web management

The SQL server database is stored on a separate secured server that is not directly accessed by any of the interviewers or respondents.

3.9 Switching modes

Important part of this is that the switching of the mode is driven by the CATI side; the web side is passive. All mode switches are initiated by batch routines, iwer actions, or supervisor actions.

Since it is the CATI mode controlling the mode switch, and SMS in control, additional Manipula routines are called at strategic points during the CATI interviewing process. One Manipula script updates the web management system database with call records, contact information, and key variables. Other Manipula scripts are responsible for setting flags in the one mode (i.e., web) to lock sections completed in the other mode (i.e., CATI) so that the respondent doesn't answer the same sections twice. Note: in case of a partial section completed, that section is restarted in the new mode.

3.9.1 Interview on demand

If a case has been assigned to web, and the respondent calls in using the toll-free phone number, then the need is to start the CATI interview immediately. This is accomplished by having all sample loaded in both web and CATI at the same time. The CATI interview may not be part of the current release, but is always available by its primary key or phone number.

The interviewer then conducts the survey as normal by going through the coverscreen and then the interview. Upon a suspended or completed case, a call record is written to both SMS and the web management system.

3.9.2 Flags for sections started/completed

To reduce respondent burden, any sections that are completed in a mode are flagged within the study data model (CATI/web). The flagged sections are then kept on the route in the other mode and are not asked. Key variables that are used in other sections of the survey are copied for those sections that are locked; this maintains the flow of the survey in the other mode. For example, if Block A is completed, and Field A within the block determines which of Block B, C, or D is asked, then Field A is a key variable that has to be passed to the other mode. Other fields within the section that locked are not copied.

Thus, suppose sections A and B were completed in web, and section C was partially done in web. Then the interview was switched to CATI. The interview then would start in section C. The data for sections A & B are only in the web interview, and there would be duplication for some fields in section C between web and CATI. Resolving conflicts between the data will be handled by analysts.

The script to determine what makes a completed section does a rules check in the data model (e.g., CATI), steps through each asked field on route, and keeps track of each block that has data. It also notes the last block with data, and keeps that off the list of blocks to flag. Once the list has been determined, flags in the other mode data model (e.g., web) are set, and the rules in the data model use the flags to determine to KEEP or ASK the section. It is possible to switch modes multiple times, and each mode then will have a unique part of the survey according to what was filled in. That means, CATI could have sections A, B, F, H, while web has sections C, D, E, G, and I completed. This scenario implies CATI: A, B; then web: C, D, E; then CATI: F; then web: G; then CATI: H; then web: I. There are possible overlaps in sections B, E, F, G, H, and I.

3.9.3 Manipula scripts – batch mode

Overnight scripts also move sample between modes. This script logs its actions to a SQL database, sets appropriate flags for holding/releasing the sample, and assigns randomized appointments according to a complex algorithm.

3.9.4 R logs onto web survey at will (mode start in CATI/web)

The respondent can complete the web survey at any time. The flags for the sections should always be up-to-date from CATI. The respondent can break off and resume the web survey. However, if the interview is switched back to CATI, tthe flags would then be set again before the CATI interview is initiated.

4. Conclusions

Any large project takes effort, and one with a mixed mode involved takes even more. The project as a whole has been successful.

4.1 Lessons learned

4.1.1 Mixed mode switch

The switching between modes was more difficult than expected in terms of how to manage the interview data between modes and keeping case status updated in the different systems (SMS and the web management system). The breakthrough was the realization that one system, SMS, needed to be in control of the status of all cases. That led to creating Maniplus scripts that set flags for locking completed sections, and for updating the status in the systems.

4.1.2 Autoindex + BOI files

The OLEDB manager wizards do a very good job of generating a data model and a matching BOI file. However, when the SQL table contains an autoindex column as a primary key, inserting records from Blaise is problematic. The solution was to use OLEDB manager and manually select the fields to create in the BOI file, excluding the autoindex column. Inserts from Blaise then work seamlessly.

4.1.3 Testing

As has been said many times before, the more time that is planned in testing yields greater confidence, data quality and fewer changes during production. As much as was desired, more time for testing was desired.

A parallel test project had not been set up initially on the production server, but instead test sample was loaded and removed once production started. If it had remained, then potentially test data could have been introduced into production sample. This meant that true testing of new releases in a production environment could not happen until a parallel test project had been established.

4.1.4 Maniplus flag setting routines

We used the Manipula meta information methods and recursion in the scripts to be able to accurately set flags in the CATI/web surveys. The point is that a survey taken in one mode had to update the other mode's survey flags.

4.1.5 Priorities

It was difficult to balance work between programming the surveys in CATI & web and working on systems development. Typically, the survey is more important – what good is it to collect bad data? In hindsight, it would have been better to work on these parts in tandem since there were system requirements that affected the data model design.

4.1.6 SMS

Setting up the project for mixed mode also required modification of the CATI sample management system. SMS had to separate from other normal CATI projects because supervisors who conducted an interview from the management program, SMSSuper, also ran the pre-coversheet and post-interview scripts to set flags.

Setting up new users in this system took extra effort because of drive mappings, dedicated computers with equipment for CARI, installed DLLs, and checking to see that all parts of the mode switch functioned.

4.1.7 Web receipt page

The receipt pages are responsible for transferring data from the main interview to a recontact interview as preload. Creating preload on-the-fly between two Blaise IS surveys will allow us to create complex new Blaise IS surveys in the future. The same receipt pages also stored key variables to the web management system via stored procedure calls written in ASP.

4.1.8 Development

Using prepare directives to produce testing & production datamodels made it easy to ensure all parts of the survey were compatible. Keeping the builds of Blaise versions current between servers was also key in reducing headaches.

4.1.9 BOI data to SQL

We discovered that user entered data that contained apostrophes, DK, or RF caused errors when being written to an existing SQL table. The single quote within a string field was interpreted by SQL as ending the string, and the rest of the string was being interpreted as SQL commands. This is the essence of a SQL injection attack. This was solved by escaping all single quotes to two single quotes.

DK and RF also were transformed by Blaise into a field of 9..9's and 9..8's. We rewrote the Manipula routine to customize the values so they fit into the data column.

4.1.10 Slow access

We found invalid references within Manipula scripts to BOI files caused a Windows seek & wait, and really slowed the execution of the script to a crawl. This happened when we moved from the testing server to the production server. Correcting the server reference solved this problem.

4.1.11 Testing documentation

Programmers and testers actually used two different specification documents. The programmer document lagged behind the testing document, and as a result true bugs were reported that did not match current programming document. In the future, one common document should be used for both.

4.2 Future directions

Updates to the SMS are necessary in order to make it easier to work with mixed mode surveys. These include adding a "mode" variable to the coverscreen, moving interviewer and contact observation variables to an external database, removing inefficiencies, maintenance, and security problems with the current system, such as shared text files.

We plan to create a "gold standard" project to use when starting a new mixed mode survey, and also have a library of groups for complex pages to help standardize the web surveys.

4.3 Insights

We want to share these insights in developing a mixed mode system:

- Spend more time reading through all the Blaise help related to mixed mode, Blaise IS, and data transferring routines before programming starts. We found that we were "going back to school" throughout the project and finding better ways of performing tasks, and sometimes rewrote code to take advantage of our new knowledge.

- With limited time constraints, we recommend producing parts of the management system as feasible. It still is very important to have a well-formed survey for data collection, but the systems have to be developed at the same time in an iterative approach.

- There were only two programmers, with temporary assistance from a third, in programming the CATI and web surveys and the SMS/Blaise IS management systems. One other programmer developed the SQL web management system. It would have been extremely helpful if there had been a programmer for each of the parts: one for CATI, one for web, one for SMS/Blaise IS management systems, and one for the SQL web management system.

- And finally, for all those non-programmers: Meetings are most useful when they are short and concise as this will allow more time for the actual work than talking about the work.

5. Footnotes

[1] SurveyTrak is the University of Michigan case management system for distributed sample, i.e., field data collection.

[2] Coverscreen datamodel: see 1.3

[3] Study datamodel: The actual survey itself without any management piece embedded in the survey. There are standard fields, such as Complete, that are required for SMS and SurveyTrak.

[4] Result codes are call dispositions, such as an answering machine with a message left is code 1402.

[5] Daybatch sub-priority: a more recent feature of Blaise CATI, is a sub priority code within a result code to influence the delivery of a case.