Lifecycle Processes to Insure Quality of Blaise Interview Data

Linda Gowen and Pat Clark, Westat, Inc.

1. Introduction

Efficient, accurate and timely processing of interview data post-collection is critical in both cross sectional and longitudinal survey environments. Through the use of the Blaise API a comprehensive approach has been implemented for the processing of Blaise interview data through the entire survey lifecycle. The Blaise API makes it possible to use 'add-on' tools and processes at each of the survey lifecycle stages to insure data quality. This paper reviews some of the tools used at Westat with a focus on the dynamics of managing processes impacting data quality. The experiences implementing a lifecycle approach for COTS Blaise studies across multiple survey environments within an organization are discussed.

2. Survey Lifecycle Overview

Tools have been developed to support each of the major lifecycle stages in a COTS Blaise study including design, development, data collection and post-collection processing, and data delivery. At the start of each study, a core project team with the major stakeholders (Client, Project Director, Systems Manager, and Data Manager) decides which tools to utilize in conjunction with the particular study. The major lifecycle stages are as follows:

I. Design
   - Requirements Analysis
   - Develop CAI Specification

II. Development
   - Develop Blaise System
   - Internally Test Blaise System.
   - Pretest for Field Readiness

III. Data Collection
   - Deploy Blaise System

IV. Backend/Data Preparation
   - Edit and QA Blaise Data
   - Create Codebooks

V. Data Delivery
   - Deliver Data
   - Data Documentation

Figure 1 shows the major tasks in each of the stages and the dynamic interplay between stages throughout the lifecycle. Blaise provides many system capabilities used to integrate external tools and processes including:

- Blaise API – permits use of relational databases and data manipulation through other programming languages such as Visual Basic
- Manipula - batch system to import, export, and recode data that can be used for reporting and data manipulation
- Lookup files can be incorporated into the design as separate databases and can be searched to prevent misspellings and to facilitate text entry during interview
- External data files can be in ASCII format

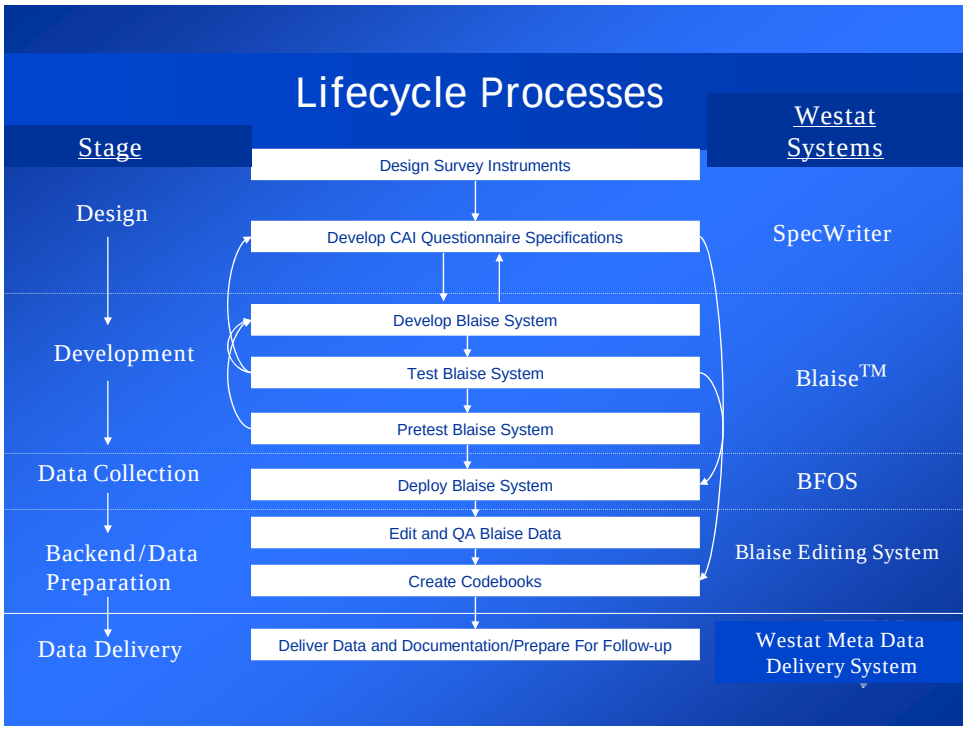

Figure 1: Lifecycle Processes

2.1 Stage 1 – Requirements Analysis

The goal of this stage is to define both the Blaise and non-Blaise requirements for the system. Since one Blaise system is never a stand alone system, it is necessary for the Systems Manager to prepare a project data flow diagram. It must show all sources providing input data to the all Blaise Systems, as well as, the data required to feed external sources. The details of external system integration should be designed and incorporated into the CAI specifications. The data flow diagram will provide a detailed illustration of how all data are input and output by the Blaise system. Simply put, the flowchart should illustrate how the individual questionnaire data will

move through all systems needed for processing and how the different systems work together.

2.2 Stage 2 – Develop CAI Specifications

In this stage the CAI specifications document will be created. This document will include the "complete" hard copy questionnaire typically needed for regulatory compliance with additional detailed instructions necessary for programming the CAI system.

Walkthroughs with the entire team are an important component of the specification development process. All perspectives need to be considered including the research objectives and costs of various approaches. In the paper-and-pencil environment, special cases were resolved by an interviewer writing marginal notes. In CAI, these situations need to be anticipated in the specifications if special question sequences need to be programmed. In CAI interviewers can always enter comments, but like marginal notes these comments require special review post-collection in order to assure data quality. These types of considerations should be addressed in the CAI specification development process.

We use a tool called SpecWriter, to help capture the hard copy content and programming specifications for the instrument. It is a front-end application built to be a survey development tool as well as capture details needed for programming. Its purpose is to integrate the project design and systems development of survey questionnaires. The tool is comprehensive and provides the ability for all team members to make their contributions in one location. The study staff can design and specify survey question-and-answer sequences, logic checks, and question routing, while data management and systems staff can use it to add detailed logics; edit formulas and information about the Blaise structure.

SpecWriter facilitates versioning by tracking changes to the instrument. Changes are made easily in the system all the while documenting and revising specifications. Each definitive version of the instrument can be identified with a version number. The version number assigned to the instrument, flow-chart and hard-copy questionnaire should be consistent so that it is clear what relationship each has to the other. One can also use version numbers to control and track changes, maintaining a log listing the changes made to each. Logging with versioning is useful in the development and maintenance of the CAI instrument because it provides a history of decisions along with resulting changes. This helps to avoid errors that may be repeated given the complexities in the particulars of a project.

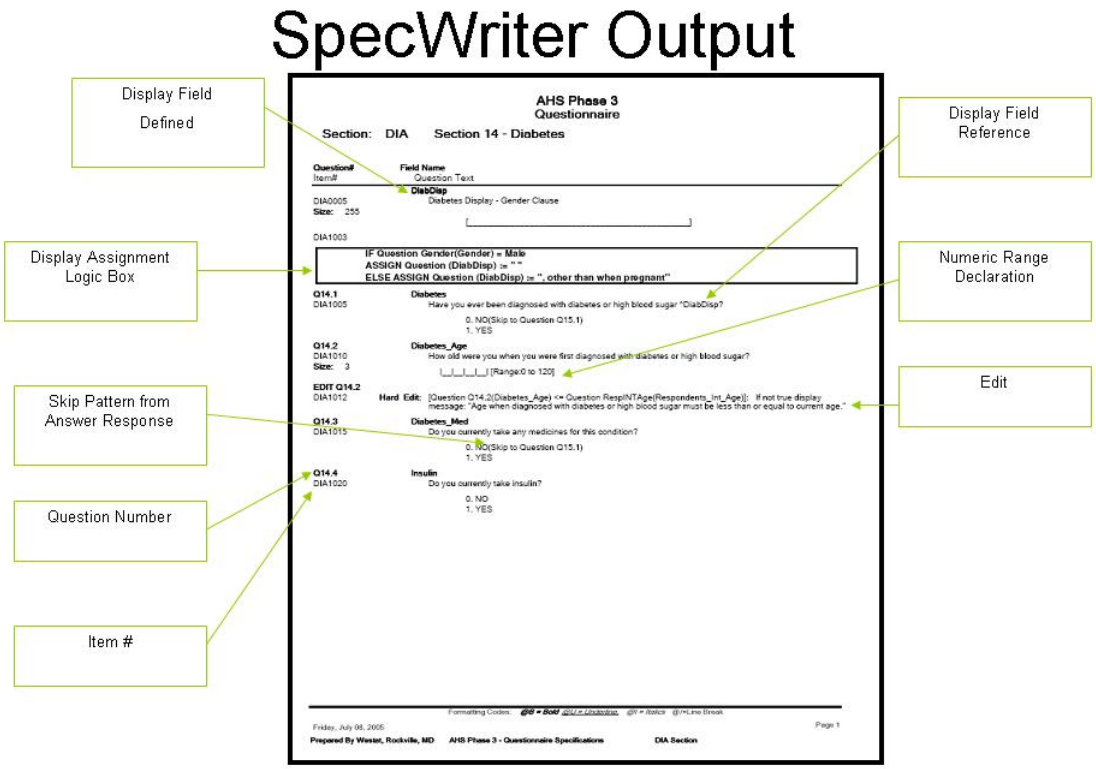

Figure 2: SpecWriter Report

SpecWriter creates several version controlled reports with varying levels of detail for use by: clients, project managers, data managers, and programmers. The ability to present specifications at varying levels of specificity helps to bridge the language gap between programmers and study managers. Another benefit of using SpecWriter is that it can create programming code for inclusion into the Blaise system.

2.3 Stage 3 – Blaise Development

Programmers use the output from SpecWriter to begin source code development, and the data flow diagram in the assembly of all components of the overall system. An important consideration for Blaise system development is to know how Blaise stores the data in conjunction with the field and block declarations. Also, it is essential to know how data are extracted out of the system. Testing should include extraction procedures by checking the creation of output files and variables. Navigation and repeating blocks need careful review to avoid data overwriting situations. It may be necessary for programming to include passing data between blocks for explicit creation of foreign keys.

Key considerations to assure that the collected data can be properly transformed for data delivery at the end of the data collection period include:

1. Proper use of data to be used as foreign keys in the creation of blocks/tables as the instrument is coded

2. Review of sampling or randomization processes being coded

3. Testing and QC at all stages of instrument development including verification of the data integrity after both forward and backward navigation

4. Designation of field name lengths appropriately for later SAS processing;

5. Relationship of question numbers to Blaise variable names, and

6. Ensuring that all needed intermediate variables are saved

During production and post-processing, Blaise data may be needed by other processing systems. Advance planning for reporting is considered so the programming staff can assure the database structure provides for the easy extraction of the needed information, or that programs are developed to permit viewing within the Blaise database itself. Timing variables and status variables should be included in the code to support reporting. These requirements should be captured in the CAI specifications document.

2.4 Stage 4 - Internally Testing the Blaise System

A specialized testing team ensures a level of quality in systems and data that has become more difficult to attain as the infrastructure becomes more complex. Testing is done using all systems that will be functioning in the field or in the home office. Using the business requirements, questionnaire specifications, and specifically designed 'test' cases, allows for thorough testing of each instrument. All testing is based on workflow with regard to how users will do their jobs, based on all the provided requirements and specifications. Functional testing is the main focus, which is to verify that each system conforms to all requirements as stated in the specifications. Feature by feature validation is performed, making sure that all skip patterns work properly, and using boundary values and erroneous input data to check error messages. System documentation such as user guides and training materials are tested to ensure accuracy.

As with any software development effort, it is important to track problems and their resolutions when developing a Blaise instrument. This ensures that all problems have been noted, and either resolved or had a decision made in regards to it. Track Integrity, an off-the-shelf product, provides robust capability for tracking problems including numerous customizations features. Users are able to configure the system to meet specific requirements in their problem tracking and resolution process, including the option to add user-defined variables.

As tests are run, new requirements may emerge based on issues reported in testing. The schedule may be adjusted accordingly, test cases are updated, and decisions on

all issues are recorded in the issue tracker. Final testing approval is measured by successful completion of all test cases and by the closeout of all issues. Although testing approval may have been received on the fielded instruments, any subsequent changes made to those instruments require additional "regression" testing. Regression testing includes the testing of all current valid test cases as well as new ones resulting from the new requirements or fixes. This type of testing is essential to quality assurance because it eliminates the possibility of introducing unexpected errors to previously working functions. Regression testing, particularly on a longitudinal study, is an integral part of quality assurance throughout the lifetime of the study.

2.5 Stage 5 - Pretest for Field Readiness

While specialized testing is an important aspect of an instrument's lifecycle, it is important that testing is also performed by the project, data management staff, and field staff. It is vital to identify idiosyncrasies that will need to be dealt with in training or in post-collection activities.

A good place to start is in constructing the mock data for the role plays that will be used in training. These role plays should simulate data collection scenarios. The goal is to have each condition tested for all the responses available in the instrument, to check that the appropriate paths are followed. An automated training script system, WinCATS, has been used in designing role plays. It creates screen captures of the Blaise instrument and annotates the document with imported graphic images designed to replicate the question/response flow of an interview. The electronic format allows for easy updating when changes are made to the collection instrument. Once the mock data has been constructed, the full cycle of the system is tested from input through delivery. Not only does this give confidence in the functioning of the questions, but it allows the team to pinpoint questions where specialized coding will be needed and areas that might need emphasis during training. Files with the delivery layout are built from the mock data and frequencies run to verify that the system is capturing the data correctly.

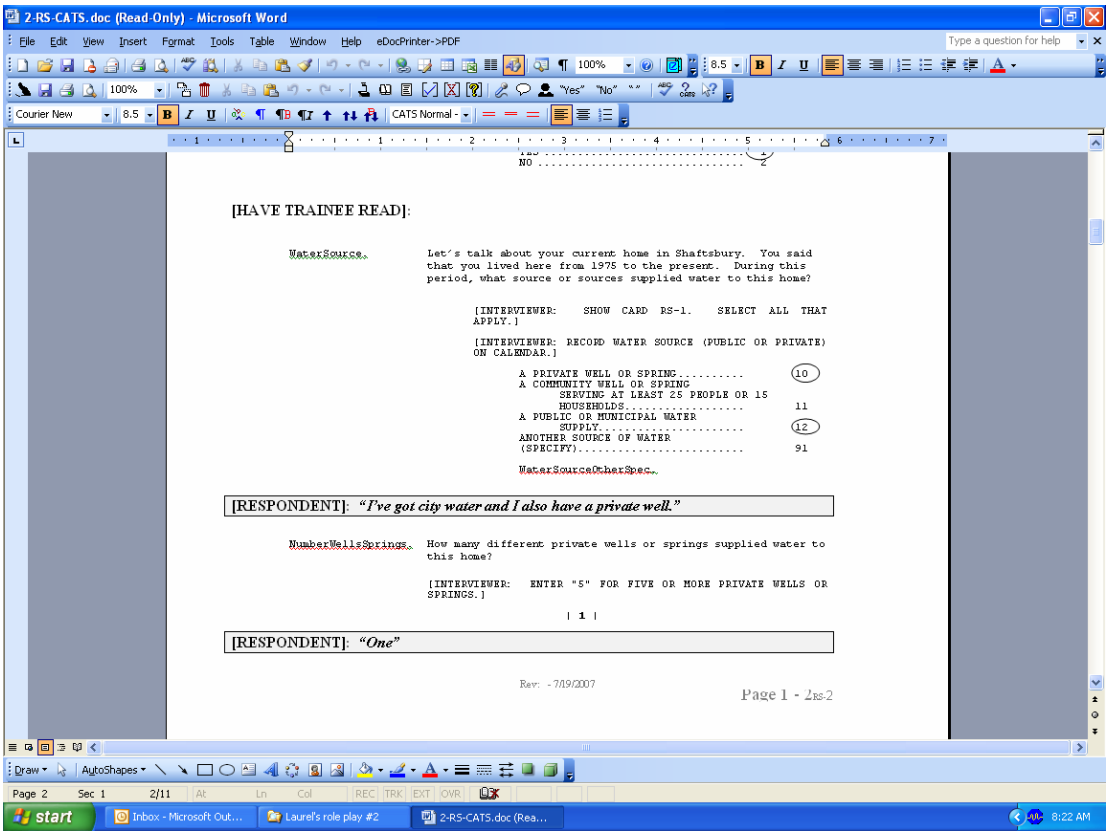

Figure 3: Example of WINCATS role play document.

2.6 Stage 6 – Blaise System Deployment

Deploying Blaise systems to the field can vary in complexity depending on the suite of programs needed for the management and distribution of the Blaise software and its accompanying data. Although projects vary in their implementation, all projects have field management or telephone center management systems supporting the Blaise applications. Westat's Basic Field Operations System (BFOS) provides CAI projects with a baseline field management system designed around a field model common to most Westat field data collection projects. In addition, BFOS can be adapted and customized to meet more specialized project needs.

More specifically, BFOS provides projects with the following:

- **Interviewer Management System** - A laptop-based application that allows interviewers to review and change assigned case status, launches the Blaise CAPI instrument, and synchronizes management and CAPI data with the home office.

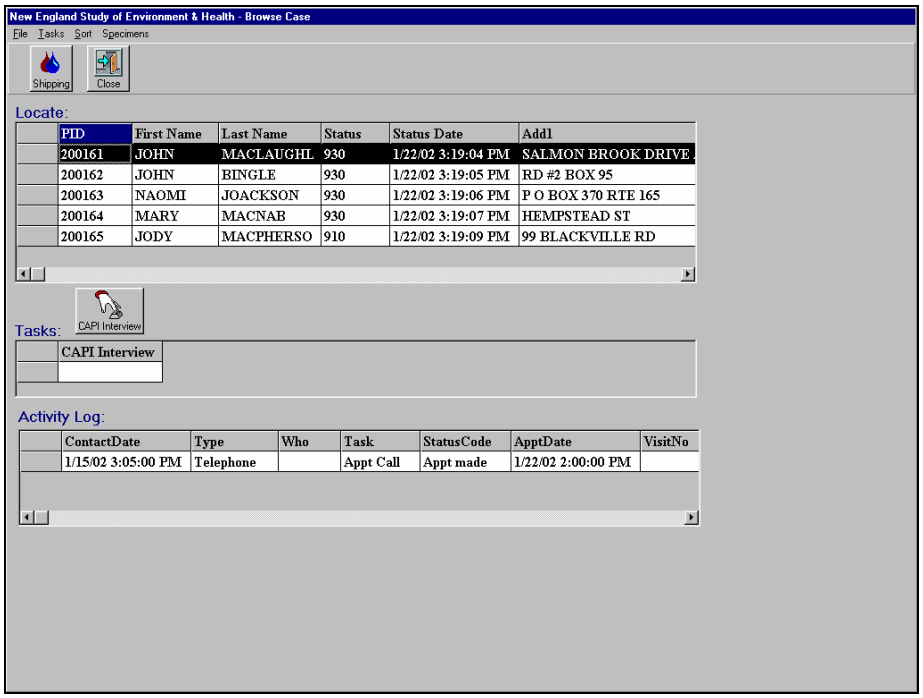

Figure 4: BFOS Case Browser

- **Supervisor Management System -** This is the web-based application used by field supervisors to view and manage the work of interviewers, including case assignment and transfer between interviewers, review and adjustment of case status, and other administrative and reporting functions.

- **Home Office Management System -** This provides functions needed by home office staff to load and manage the study sample information and to maintain information about the field staff and their assignments.

A very important aspect to the quality of a Blaise system is to closely track when the different versions of a Blaise application are deployed. Each new data model should always include a version field that is changed to reflect the field release number. This helps with troubleshooting and can be used for documenting in a codebook. The version can be used in codebook inapplicable statements that document why a field is not answered. The version field can be used to identify when a field was introduced in a subsequent release. The practice of setting the version field to the new value will force a new version of the Blaise data model. This is good because it forces a conversion from the previous model to the new model which will force the project to address differences in the data early in the process. Also the deployment process should include logging of dates and times when particular versions were implemented.

2.7 Stage 7 - Editing Blaise Data

A Blaise study does not end with Data Collection. The completed interviews are scrutinized for unusual entries, all comments are reviewed for consistency with the data entered, and all derived variables are created before processing for data delivery

is completed.

An array of systems supporting the Blaise editing will be implemented, including:

- Extract system
- Editing system
- Comment system
- Data Decision Log
- Data and interview validation
- QA reconciliation systems
- Study management systems
- Metadata system

Collected data must be made available, both for reviewing and processing. The extract system allows the data manager to access all of the data for all cases. Extraction may be performed through Manipula or Blaise API.

In the Editing process, data are reviewed and prepared for delivery. The editing system, developed in Blaise, allows the data manager to see the case data in the interview format. It displays one case at a time. The Editing System data are maintained separately from the unedited Blaise database. Decisions about delivery should be considered when initializing the Editing system which can replicate what is currently being collected in the field with the addition of derived variables, additional edits, and updates as needed.

The project flow chart will contain actions specific to the review and editing of the data. Basic steps of the editing process are:

- Combine all of the data into a master database that can be seen by the data editors,
- Maintain the Blaise interview(s), viewable in both interviewing and data editing modes,
- Display comments collected by the interview in a comment report, as well as in the interview itself,
- Data editors determine which fields may need data changed based on comments from the interviewer,
- If the data editor makes a change, the corrected skip patterns will be enforced,
- All decisions will be recorded in the data decision log,
- Other, specify fields will be coded,

- Frequencies will be reviewed, and

- Reconciliation with other forms or systems will be completed

Comment review is a major part of our editing process. Comments allow the interviewer to communicate information about particular questions that may impact the answer. The ability to view all responses associated with a comment allows the editor to follow the thought process that led the respondent and interviewer to the reviewed response. Based on the review, and following previously established coding rules, a decision is made as to whether the data entered during the interview needs to be updated or whether the remark is consistent with what was entered.

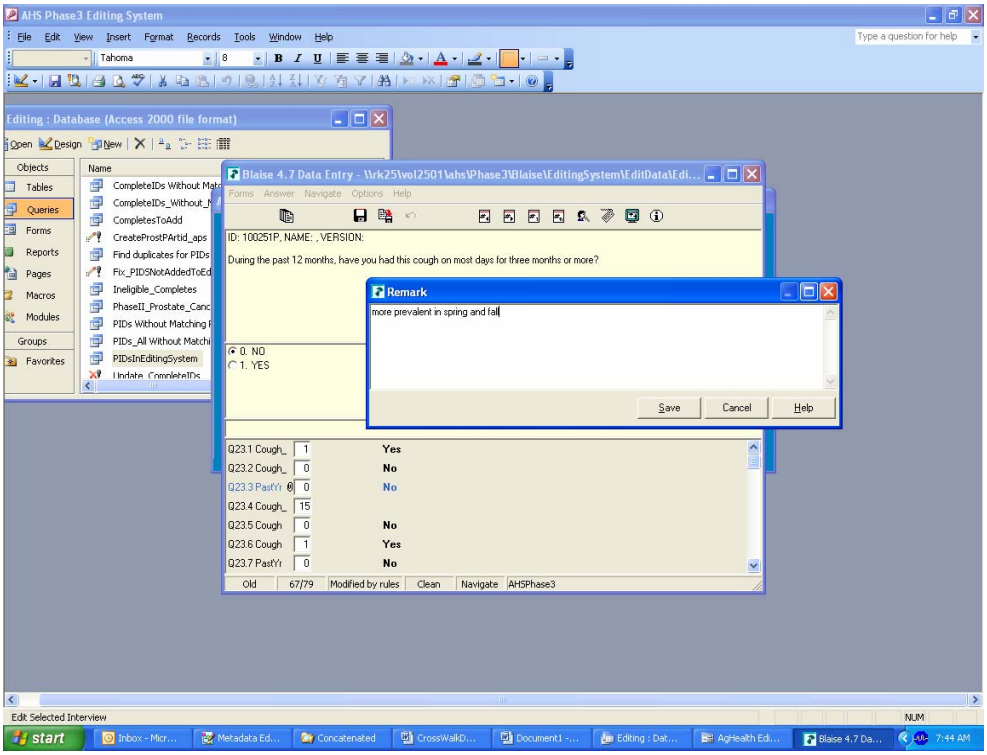

Figure 5: Comment Review

In one recent study, 36% of the interviews had at least one comment. After reviewing all comments and following the editing rules, 23% of those comments had some type of change made to the data.

Special attention has been given to the creation of a multi-faceted system which pulls the comments from the extracted data, links to the editing system, and allows for the documentation of decisions about those comments and any other data changes that are deemed necessary during the editing of the interview. The Data Decision Log is used as a tool to monitor completeness and consistency of the coding and review effort and as documentation for any decisions about the data that were made during the data processing effort. Extra fields have been added to allow for the tracking of the process, to allow for better time management.

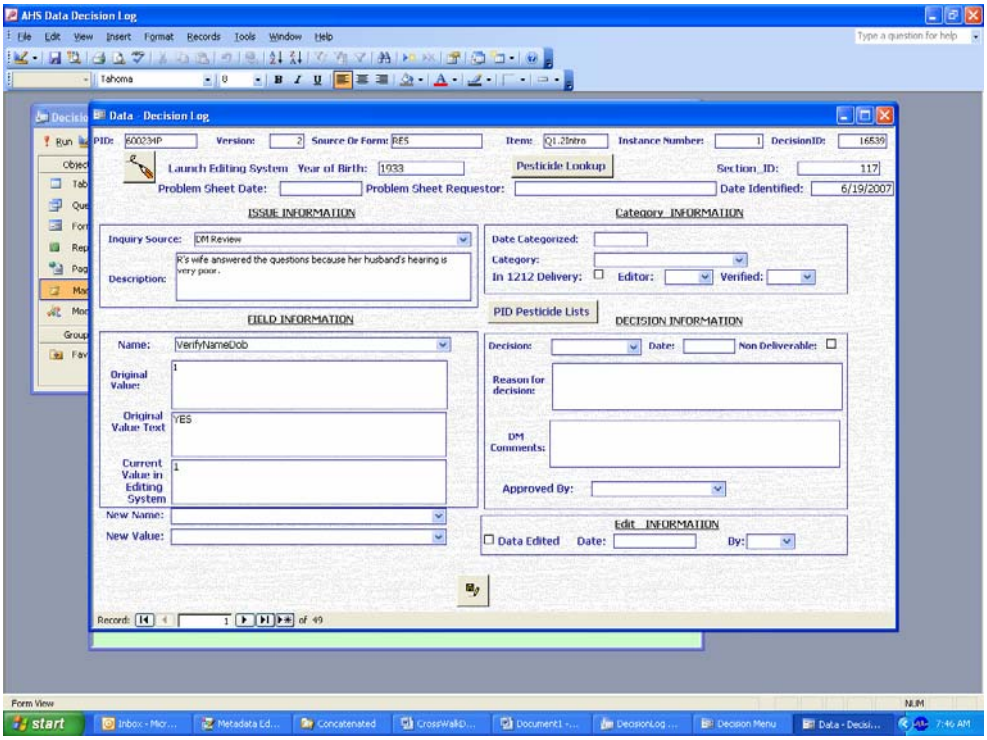

Figure 6: Inside the Data Decision Log System

All verbatim entries are dealt with. Depending on client requests, other specify fields can be re-coded into the existing code list, coded into an expanded code list, or coded as a separate variable. Specialized coding can be done manually inside the editing system or linked to it from an outside system, and completed either manually or automatically depending on the system. We have found it helpful to have separate reports for all non-numeric open-ended fields.

As a final check of the data files themselves, the data are extracted from the final edited tables and frequency reviews are performed.

There are often redundant data that are recorded in many data sources of a project. Our QA system was designed to report the discrepancies in redundant data. For example, biological sample statuses could be recorded in the Blaise Interview, the field management system, on Hardcopy forms, as well as in the Biological Repository Inventory. All of these data sources need to be reconciled. The QA system is a secured system that contains various reports based on the Editing system data, the Study Management System data, and other project source data for reconciliation. The system also has the capability of logging errors which can be suppressed on the reports. The documentation of the resolutions is concatenated with the Data Decision Log.

2.8 Stage 8 - Create Codebooks

An important part of documentation, both for internal use and for delivery, is the codebook. Each instrument requires documentation. Our codebook generation system was designed around the metadata output from the Blaise instrument. Within

the codebook generation system, one can:

- change the order of fields;
- modify question numbers, question text, and allowed values;
- add questions;
- add explanatory notes;
- add documentation of skip patterns; and
- add logic specification boxes.

Each data element is versioned, which allows for continuity of collection. All derived data elements have the associated derivation details presented in notes. Once the information is in place, one can generate a codebook without frequencies, a codebook with frequencies, or a data dictionary.

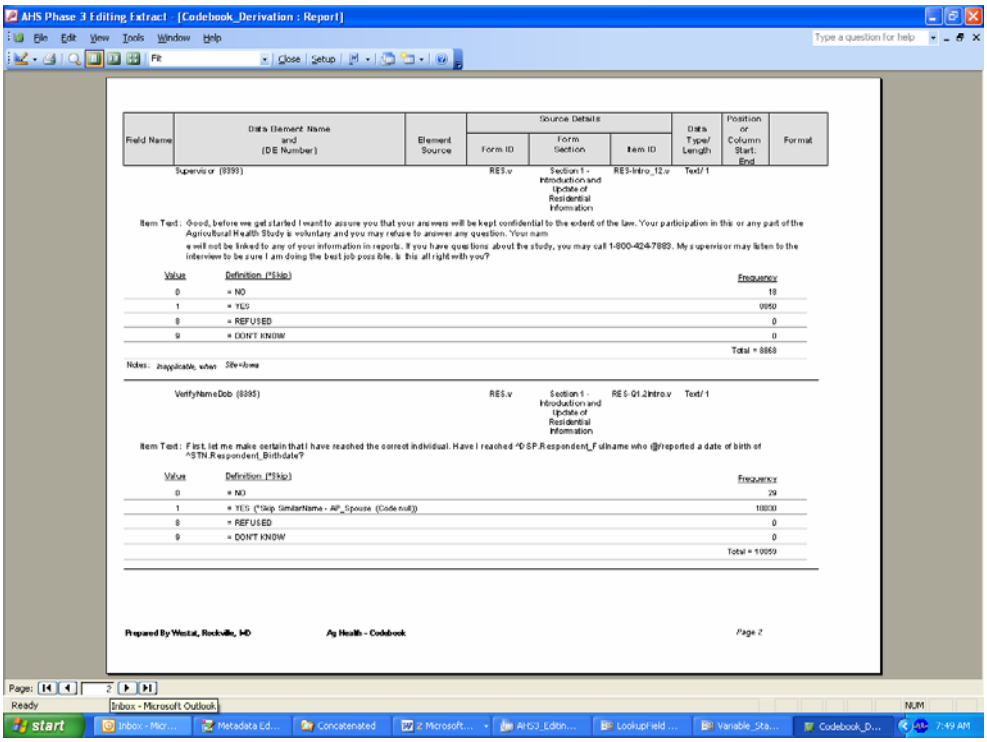

Figure 7: Codebook with Frequencies for Delivery

2.9 Stage 9 - Data Delivery

Once the data are reviewed, reconciled, and documented, delivery of both data and metadata commences. Westat has developed a Data Delivery Metadata System which supports delivery of metadata associated with datasets such as raw data, research datasets, analytic datasets, restricted use datasets, and public use files. The

system is designed to provide descriptive information about the context, quality and condition and characteristics of the data. This information helps one to understand how the data were collected and any post-collection handling that might have affected its use and interpretation. Authorized users are allowed to search, view, print, or download the metadata for which they are given access. It allows users to identify commonalities across studies, which may otherwise go unnoticed. The web-based system allows for multi-user access with strictly controlled study-specific user privileges. It is planned to remain compatible with emerging industry standards for integration and possible harmonization with external systems, and will accommodate interfaces based on common, open standards.

Projects may require only one delivery or multiple deliveries. Longitudinal studies routinely require annual, quarterly, or monthly deliveries. Complete documentation of each delivery allows for standard outputs, integrated version control, searchable linked metadata, and archival storage and tracking of metadata files.

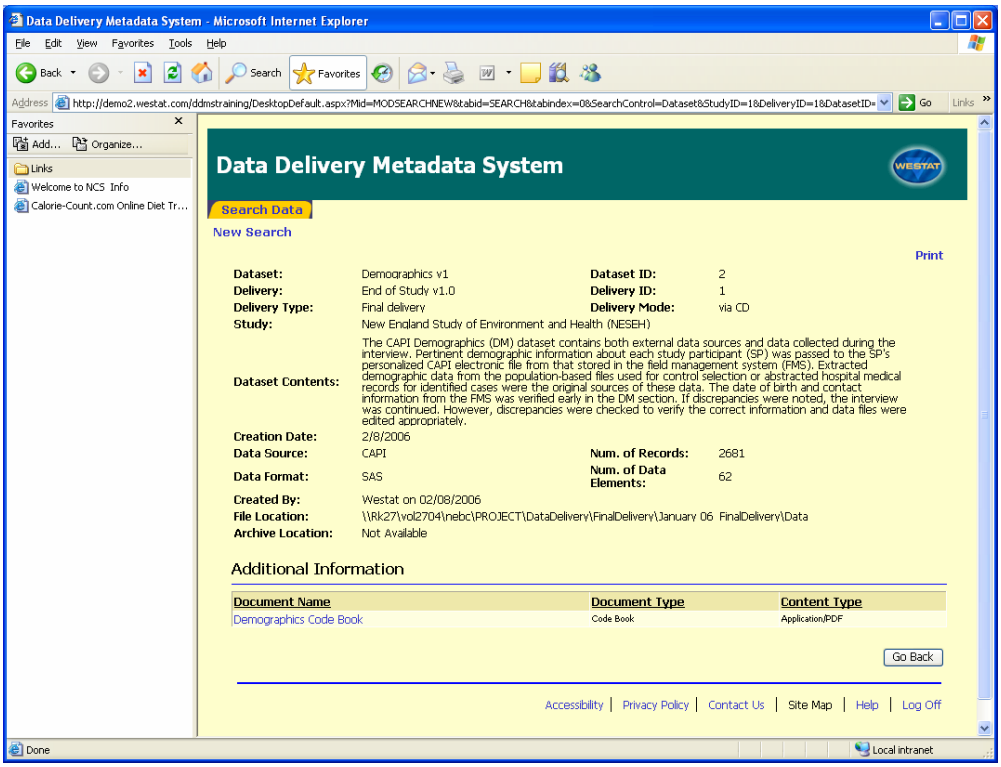

Figure 8: Data Delivery Metadata System

3. Summary

We have found that managing aspects of the processing extending across multiple lifecycle stages is critical to data quality. Feedback from all aspects of the survey need to be considered and iterative processing loops are necessary. Lifecycle tools can be implemented in multiple project environments and customized to the needs of individual surveys. The Blaise API facilitates the incorporation of lifecycle tools. Systems which emphasize ease of use and documentation capabilities allow data processing to adjust to changing needs of a study, whether internally or externally

driven. Each step of the study lifecycle is reviewed, and a system or systems chosen to match the needs of that step. This allows a specific focus and the means to identify problem areas for each process. Active participation from all team members assure that differing perspectives are represented. Quality control is a fundamental focus of each lifecycle stage and is built into the processing operations as all systems used have components relating to data quality. In this way, quality control becomes continuous from planning through delivery, ensuring accurate and efficient data collection and processing.