Use of Blaise™ in the National Health and Nutrition Examination Survey

David Hill Westat
Christina Kass Westat
Debra Reed-Gillette CDC/NCHS
Lewis Berman CDC/NCHS

Abstract:

This paper provides an overview of the technical highlights of integrating Blaise™ instruments in the National Health and Nutrition Examination Survey (NHANES), conducted by the Centers for Disease Control and Prevention (CDC)/National Center for Health Statistics (NCHS). The current NHANES interviews and examines 5,000 people each year in 15 different locations nationwide. The survey is conducted during an in-home personal interview, using Blaise™, and a physical examination in a Mobile Examination Center (MEC). Due to the complexity of NHANES and the need for rapid data dissemination, timely quality control and ongoing field intervention, Blaise™ was applied to support personal interviews in the home and in the MEC. In addition, instrumentation and systems must accommodate change to include new research objectives occurring throughout the course of the survey. Techniques applied to meet these requirements will be described in this paper.

Technical highlights that will be described include:

- Integration of Blaise™ applications in a Windows survey management environment, on both portable Fujitsu pentops and networked examination workstations.
- Application of Cameleon™ and Manipula™ techniques for conversion of "as collected" Blaise™ data to analysis and relational forms in post-processing.
- Extraction of operation-critical data while retaining other data in native Blaise™ format and storing it in a relational database.
- Application of Blaise™ metadata and tools to manage change.

Keywords:

Pentop, Application Integration, Data Availability, Relational Database Integration, Data Management

1. Introduction

This paper describes the NHANES pentop based computing environment used for household interviewing operations, use of Blaise™ to conduct Audio Computer Assisted Self Interviewing (ACASI), and technical methods used to deal with critical data management and operations challenges involving Blaise™ instrumentation on the NHANES. These methods were designed involving combinations of Blaise™ utilities and external tools to directly address the data management and operations challenges of the project.

One particular Blaise™ method applied extensively was the use of the Cameleon™ utility. This utility was used for generation of data documentation from the Blaise™ metadata, and to promote the data resulting from multiple Blaise™ data model versions into a coherent single and continually changing data representation for the study. The latest releases of Blaise™ with Open Blaise™ Architecture (OBA) additions will provide functionality that may ultimately replace these Cameleon™ methods.

2. An Introduction to the National Health and Nutrition Examination Survey (NHANES)

The National Health and Nutrition Examination Survey (NHANES) is one of several surveys conducted by the Centers for Disease Control and Prevention (CDC) which assess the health status of the United States population. This program is an ongoing series of surveys that originated in 1960 as the National Health Examination Survey (NHES), Cycle 1. Since the start of the program, there have been eight periodic surveys[*]. Beginning in 1970, during development of the fourth study, a nutrition component was added, and the name was changed to NHANES.

The primary objective of NHANES is to collect high quality health and nutrition data and to release it for public use in a timely manner. In accordance with this objective, NHANES has the following goals:

- To estimate the number and percent of persons in the U.S. population and in designated subgroups with selected health conditions and risk factors;
- To monitor trends in the prevalence, awareness, treatment, and control of selected diseases;
- To monitor trends in risk behaviors and environmental exposures;
- To analyze risk factors for selected diseases;
- To study the relationship between diet, nutrition, and health;
- To explore emerging public health issues and new technologies;
- To establish a national probability sample of genetic material for future genetic research; and
- To establish and maintain a national probability sample of baseline information on health and nutritional status.

NHANES data have been used for numerous purposes. Some examples include:

- To determine the prevalence of iron deficiency in the U.S. population[1];
- To estimate the prevalence of osteoporosis in the older U.S. population[2];
- To examine the prevalence of overweight among U.S. preschool children[3]; and
- To determine the consistency between U.S. dietary fat intake and serum total cholesterol concentrations[4].

The diverse nature of NHANES data attracts a wide variety of users who evaluate health issues and develop policy and set national health priorities. Some of the primary users of NHANES and NHES data have been researchers in public health. The richness of the NHANES datasets extends the usage beyond the public health community to include industry, other government agencies, policy makers, and students. While the NHANES data are particularly important for evaluating the nation's health, it is also being used for information technology research and development.

The most recent survey differs in several respects from previous surveys in this series as follows:

- The current NHANES is continuous, whereas previous surveys operated in a fixed time frame.
- Survey content can be changed from year to year, depending on the complexity, reliability, and validity of the health measure, scientific advances in testing, newly developed protocols, and new equipment.

---

[*] NHES I (1960-62), NHES II (1963-65), NHES III (1966-70), NHANES I (1971-1975), NHANES II (1976-80), HHANES [Hispanic population] (1982-1984), NHANES III (1988-94), the current NHANES (1999-continuing).

- The current NHANES features incorporate and rely on information technology (IT) to collect data and insure high quality. IT features include a private frame-relay wide-area network, commercial off the shelf (COTS) equipment, local area networking, relational database technology, a data replication architecture, development environments such as Blaise™ [6] and PowerBuilder™ [7], and integration of biomedical equipment. The technology innovations in the current NHANES result in real-time data access, rapid and accurate data collection, reduced back-end data cleanup, and faster data release and analysis.

Earlier cycles of the study relied on paper recording methods in booklets for both interviewing responses and examination results, with subsequent keypunch data input. With advances in the portability of computer equipment, the study embraced real-time field data entry of examination results starting with NHANES III (in 1988). Mid-way through this survey cycle (1991), adequate laptop computer and software were also available so the questionnaire portions of the study were converted for computer assisted personal interview (CAPI) data collection[8]. For the present survey cycle, IT advances were fully incorporated into the implementation of a supporting, fully integrated, information management system for the study.

The vision for the current NHANES was a system that verified data at the point of entry, contained shared common data elements, reduced the burden on the participant, required minimal back-end data reconciliation, and provided portability of data across surveys. This vision led to the development of a real time data collection system. It was also envisioned that the use of COTS products (such as Blaise™) and biomedical machines with electronic interfaces would enable the data to undergo a more timely QA/QC process, enabling staff to intervene earlier and correct problems in the survey.

All these information system requirements were incorporated in the design of the Integrated Survey Information System (ISIS) for the current survey. ISIS includes the entire software, hardware, database, and network architecture. Essentially, ISIS is a collection of customized subsystems linking the field office, the MEC, contractors, and NCHS during field operations.

ISIS offers state of the art capabilities. Some of the technologies include a private wide area network (WAN); an integrated electronic data dictionary and metadata; client-server technology; data replication; and built-in disaster recovery. The ISIS infrastructure can support changes in the survey requirements to reflect changing public interest and priorities, if necessary. Most importantly, compared with previous surveys, ISIS allows NCHS to reduce significantly the time required to transfer, process, and publish the data. The impact of all of this technology was the production of edited datasets and documentation to be used for quality control analysis and for validation of health indicators within six months after the end of the first year of the survey. By comparison, it took three and one half years to produce public use datasets from NHANES III.

Each year, NHANES visits 15 different sites, with operations lasting about eight weeks at each site. The NHANES operation uses three mobile examination centers (MEC), each of which consists of four interconnected tractor-trailers that move to sites around the country. Each site constitutes a primary sampling unit for the study where approximately 1,000 people are interviewed in the household, yielding 300 to 500 participants selected and examined in the Mobile Examination Center (MEC).

At each site, significant advanced planning and arrangements lead to the opening of a local field office and subsequent arrival of the Mobile Examination Centers. A team of approximately 20 interviewers arrives at the site to start the screening and interviewing process. When the interviewer arrives at a sampled household, he or she shows an official identification and briefly explains the purpose of the survey. The interviewer then administers the household screener questionnaire solely to determine if people in the home are eligible to participate in the survey based on pre-determined demographic criteria. If the person is eligible, the interviewer explains the household questionnaire and informs the participant of their rights and the CDC/NCHS confidentiality policy. If the person agrees to participate, the household interview is administered. Upon completion of the interview, the interviewer schedules a time for the participant(s) to be examined at the MEC.

The household questionnaire consists of three parts: screening, family interview and sample person interview. The questions asked of each participant depend on the participant's age and gender. These questions are related to health practices and experiences and in general are related to the medical components administered in the MEC. The family questionnaire portion is conducted with a designated family reference person; it includes information that applies to all members within the family. The queries elicit information concerning family structure, income, food security, and housing characteristics. Table 2.1 provides an overview of the content of each of the household administered questionnaires.

Table 2.1, Screener and Household Questionnaires.

Component	Location of Interview	Information Collected
Screener	Household	Household composition, identification of families, family relationships, demographics, food security, and Sample Person Identification.
Family Questionnaire	Household	Demographics, dust collection, family income, food security, health insurance, housing characteristics, interview management, lead dust observation, pesticide exposure and smoking.
Sample Person Interview	Household	Acculturation, audiometry, blood pressure, cardiovascular disease, demographics, dermatology, diabetes, diet behavior and nutrition, dietary supplements and medicines, digit symbol substitution, early childhood, hospital utilization, immunization, kidney conditions, medical conditions, miscellaneous pain, occupation, oral health, osteoporosis, physical activity, physical functioning, respiratory health and disease, social support, smoking and tobacco use, tuberculosis, vision, weight history

A set of exclusion criteria questions is incorporated into the household interview to determine an individual's eligibility for certain MEC components. For example, if an individual has indicated a prior heart attack in the medical conditions section of the questionnaire, that individual would be excluded from the cardiovascular fitness component within the MEC. These exclusionary data items are "shared" across the components within the MEC and are accessed by the MEC component sub-systems at the time of the exam. Not only does this practice reduce respondent burden by eliminating the need to re-ask the same or similar questions, but it also insures data consistency across components.

Upon the arrival of a survey participant at the MEC, a coordinator greets the participant, verifies basic demographic information (age, gender, and date of birth), and directs participants to the appropriate exam rooms where each examination component is conducted. Sixteen distinct examinations can be conducted, although an individual participant may only receive a subset of examinations. Two of these components, the Self-Interview and Personal Interview are conducted with Blaise™ questionnaire instruments. The Self-Interview utilizes customized DLL extensions of the Blaise™ instrumentation for the ACASI utilization of audio-recorded questions and presentation of pictures.

Table 2.2, Examination Components in the current NHANES

Component	Location	Measurements
Anthropometry	MEC/Home Exam (limited)	Height, weight, circumferences (calf, thigh, arm, abdomen) and skin folds (triceps and sub-scapular).
Audiometry	MEC	Otoscopy, tympanometry and pure tone audiometry
Balance	MEC	A modified Romberg test of standing balance on firm and compliant support surfaces.
Bioelectrical impedance analysis (BIA)	MEC	Total body water, body cell mass and fat-free body mass using a low-level electric charge.
Cardiovascular Fitness	MEC	The component uses a treadmill (walking only) to take a survey participant to approximately 80% of maximal heart rate (MHR) to measure oxygen consumption.
Dietary Recall	MEC	Detailed information on all foods and beverages consumed during the previous 24-hour period (midnight to midnight).
Dual Energy X-ray Absorptiometry (DXA)	MEC	Bone mineral content, bone mineral density, total body fat and lean muscle mass.
Laboratory	MEC/Home Exam (limited)	Sexually transmitted diseases, latex allergy, DNA, insulin/C-peptide, glycohemoglobin, glucose, lipids, nutritional biochemistries, complete blood count and many laboratory analytes.
Lower Extremity Disease	MEC	Foot abnormalities, evaluation of touch pressure sensation, ankle brachial systolic pressure (ABPI).
MEC Audio Computer Aided Self Interview (ACASI)	MEC	Alcohol use, drug use, mental health, sexual behavior, smoking, and tobacco use.
MEC Computer Aided Personal Interview (CAPI)	MEC	Alcohol use, current health status, kidney function, physical activity, reproductive health, and tobacco use.
Muscular Strength	MEC/Home Exam (timed walk only)	Timed 20-foot walk and measurement of the isokinetic strength of the right knee extensors.
Oral Health	MEC	Dental sealant, tooth count, coronal carries, orofacial traumatic injuries, dental fluorosis assessment, pain, gingival bleeding, loss of attachment and root caries.
Physicians' Exam/Blood Pressure	MEC/Home Exam	Heart rate for participants 0 - 4 years of age; radial pulse for participants 5 years and over; and blood pressure for all participants 8 years and over.
Tuberculosis Skin Testing	MEC/Home Exam	Reactions to two different TB skin tests.
Vision	MEC/Home Exam (near card only)	Distance vision, refractive error, shape of the cornea, distance eyeglass prescription and near vision.

Interviewing Operations

For the household interview and collection of self-reported information, ISIS is augmented with laptop-like computers called pentops. These pentops are programmed in Blaise™ , which allows for complex question routing, multiple language support (i.e., English and Spanish), modular design, form-like user interface design tools, and data manipulation tools. While the household interview is conducted independently of ISIS, the interviewer uploads the data to the field office server upon return to the field office. From there, the interview is replicated across the WAN back to the contractor and the NCHS servers. In addition to CAPI, ISIS also contains audio computer-assisted self-interviewing (ACASI) technology in the MEC for use in asking sensitive questions. For example, sample persons use ACASI in private via touch screen and headphones, to respond to questions about illicit use of drugs and sexual behavior. The participant can hear the questions through the headphones or read them on a screen and then enter a response, using the keyboard or a touch screen.

Metadata and Item-Naming Conventions

Increases in the variety and uses of data create a need to formalize the manner of describing both the data and their uses. Business data are created, maintained, and accessed through business processes that are implemented through applications.[9(p52)] Simplistically, metadata includes at least data about data (item metadata) and data about processes (process metadata). Process metadata includes information automatically captured in computer-aided software engineering (CASE) tools.

Collected from many disparate sources, item metadata are stored in the current NHANES database. These metadata serve a variety of functions. For instance, they allow for tracking of data changes over time, deriving new data from source data, maintaining detailed descriptions for each item, and automated generation of data documentation (i.e. – data dictionaries or codebooks). Item metadata for the current NHANES include the date at which the measurement or question is implemented, the database table the item resides in, the data type, a textual description of the item, historical information, applicable edits, the English and Spanish text of the question, instructions for asking the question, target age and gender group, response categories, skip patterns, and references to related data items. Figure 2.3 presents an example of the current NHANES metadata for one item, together with the corresponding format in a data dictionary.

Figure 2.3, Subset of the current NHANES Metadata for One Data Item and Corresponding Format in a Data Dictionary.

<table>
<tr><th>Database Item Name</th><th>Version Info</th><th>Node Sequence</th><th>Target Population</th></tr>
<tr><td>DUQ060</td><td>1.3</td><td>89</td><td>Males/females 12 – 19 years old</td></tr>
<tr><th>Database Table</th><th colspan="3">SAS Label</th></tr>
<tr><td>ANL_DUQ</td><td colspan="3">How old were you when you tried any form of cocaine, including crack or freebase for the first time?</td></tr>
<tr><th>Hard Edits</th><th>Soft Edits</th><th>Sybase Data Type</th><th>SAS Data Type</th></tr>
<tr><td>0 – 120</td><td><None></td><td>Numeric</td><td>Numeric</td></tr>
<tr><th>English Text:</th><td colspan="3">How old were you when you tried any form of cocaine, including crack or freebase for the first time?</td></tr>
<tr><th>English Instructions:</th><td colspan="3">VERBAL INSTRUCTIONS TO PARTICIPANT: Please enter an age.
ENTER AGE IN YEARS.</td></tr>
<tr><th>Response Categories:</th><td colspan="3">999 = Don't know
777 = Refused (skip to question DUQ160)</td></tr>
</table>

Such metadata require the use of consistent standards not only in the information itself, but also in the naming of items. Although the database system being used, Sybase, supports relatively long item and table names, there is a limitation on the naming conventions that can be used, since the data eventually must be integrated with statistical software packages such as SAS™ (SAS version six and earlier impose an eight-character item name length. SAS version eight allows for 32 character item-name lengths.) To accommodate in-house and external users who have not migrated to new versions of SAS or to comparable packages, the current NHANES uses an eight-character item name.

3. The Application of Blaise™ on the Current NHANES

Four distinct Blaise™ instruments are used on the current NHANES: a sample person questionnaire, a family questionnaire, a medical examination interview questionnaire, and a self-responding sensitive topic questionnaire. Each questionnaire has some unique features (for example, an extensive hierarchical structure in the family questionnaire, representing household, family, person, and event entities).

Highlights of the use of Blaise™ on the current NHANES include: the use of pentop based computers, extension of Blaise™ to implement an Audio Computer Assisted Self Interview (ACASI), integrating plug-in Blaise™ components into a Windows survey management environment, and innovative methods used for managing the data collected using Blaise™.

3.1 Use of Blaise™ on Pentop Computers

The NHANES interviewers use pentop computers for their Computer Assisted Personal Interviewing (CAPI) activities. This platform was selected to support doorstep screening with computer assistance as well as in-home extended interviewing. It was anticipated that much of the interviewer-respondent encounter would occur literally at the doorstep. Since the majority of the screeners do not yield selected participants, it was desirable to support a quick household contact, eliciting just enough information to reliably exclude non-selected households. In prior study cycles using paper data collection methods, clipboard style screener booklets were designed that could be used by interviewers when standing. Pentop computers are the portable hardware equivalent of a clipboard. They consist of a pen device on a pen-sensitive, flat display for input-output (I/O). This is ideal for encounter interviewing where a clipboard usage metaphor is ideal. Figure 3-1 shows the Blaise™ pentop environment used in doorstep screening.

Figure 3.1, Blaise™ Pentop Screening

Shifting the input from keyboard keys with the use of a mouse presents new interface opportunities and pitfalls when using pen-dependent devices. Screen space, active area sizing and placement, portions of the screen that may be covered by a user's hand, and general flow of user interaction, all become significant aspects of the user interface design. One pitfall is that while hand writing recognition is possible, the software is not adequately reliable for all users and cannot be the only entry method. Other alternatives for handling open text entries are problematic. Complete elimination of open text response in interviews is not likely since names, phone numbers, and address are all essentially open text data and typically fundamental to the screening of households.

A beneficial feature of pentop computers is that through the use of portable keyboards and a machine prop, one can easily convert them into laptop computers. Thus, pentops serve as both a stand-up platform for screening and then, when household selection occurs, a conventional CAPI laptop platform for extended interview components. These extended interview components make excellent use of Blaise™ capabilities. Figure 3-2 shows the use of the Blaise™ Pentop environment for extended interviewing.

Figure 3.2, Seated Use of Pentop Computer, Similar to Laptop Models

[Note keyboard and machine stand; user is operating pen as pointing device]

Using pentops for both doorstep screening and extended, in-home interviewing made it necessary to locate a pentop device that could support Windows operation. The Fujitsu Stylistic™ model pentops were selected to support the interviews. These devices are moderately powered Windows PCs with a pen and plug-in keyboard input devices. The Windows operating system takes care of supporting pen input anywhere that keyboard input occurs. It also handles pen taps as though they were mouse clicks. Blaise™ instruments are fully compatible, through the operating system controls, and the use of pen input does not require significant modification to the instrument software.

With advances in display specialization through the Blaise™ API or Open Blaise™ Architecture, pen friendly Blaise™ driven instruments are fully achievable.

3.2 Audio Computer Assisted Self Interviewing (ACASI) Using Blaise™

A key extension of the Blaise™ system used on the current NHANES is an Audio Computer Assisted Self-Interview (ACASI) component. This component provides the means to administer an automated questionnaire containing sensitive subject matter where participants enter responses directly into a CAPI computer; substantially reducing anticipated bias in responses when other humans (interviewers) are involved. The interview sections handled in such a manner in the current NHANES are: smoking behavior in adolescents, alcohol use, illicit drug use, kidney conditions (dealing with incontinence and erectile dysfunction), and sexual behavior.

These self-administered interviews occur in a private booth in the Mobile Examination Center (MEC). This is a well-isolated, controlled and private environment, ideal for application of this survey technology. In this setting, participants can read or listen to audio recordings of the questions (including a Spanish version). Participants enter their responses through a touch screen indicating their selections from presented responses. A great deal of methodological design went into the presentation and interaction aspects of these self-administered interviews. Some design considerations included: accommodating illiteracy and deafness, assuring accurate use of the response interface, guiding participants through the instrumentation, laying out questions for relatively simple presentation and response categories, and handling obviously erroneous or mistaken entries (soft and hard range errors). The first section of the instrument is a tutorial, explaining the operation of the ACASI system to the participant who is coached by a CAPI interviewer. The interviewer then leaves the booth for the actual administration of the questionnaire.

The implementation of this instrument is accomplished through the application program interface (API) extensions of the Blaise™ system[†] using a proprietary dynamic link library (DLL) to handle the screen presentation and response handling for self-response referencing stored audio clips of the question text and responses. The DLL routines are called through the Blaise™ API interface. The DLL acts as an external program that replaces the standard Blaise™ user interface.

The communication between the Blaise™ system engine and the external API calls is accomplished by reading the Blaise™ metadata for a question in real-time. The API interface passes the active field name (fully path qualified) to the DLL routines. The DLL routines then access the question metadata (for question text, response categories, etc.) by reading it through the same interface. One limitation of the Blaise™ version used (v.4.0) is that only metadata from the present block was accessible in real-time. Once the DLL routines have presented the question and received a response, they return the answer to the API interface for storage, and indicate completion of the question, thereby advancing the Blaise™ instrument to the next question. This process is repeated for each question on the Blaise™ route until the end of the Blaise™ questionnaire is reached.

If specialized information is needed in the external programming, an additional instrument language can be used to communicate this metadata. This is similar to the allocation of a language for help text, common in most Blaise™ instruments. In this usage, the language specific question text becomes whatever metadata (in string representation) necessary to communicate to the external routines. By judicious use of default behavior, very little need was found to add further question item specification to the ACASI instrumentation. One self-determining design characteristic of ACASI instruments reduces the need for elaborate question specification. ACASI questions by nature are relatively simple. Most respondents are likely to understand them and respond accurately.

To support the integration of audio clips, a simple default file-naming scheme was used to organize these. Each audio clip was recorded into a distinct ".wav" file named with the corresponding Blaise™ field name for question text, and appended with "_#" for each response category. The implementation included clips for each variation of question text and each response category, in both English and Spanish. Therefore, while many files are produced, using the field name to match them to the Blaise™ metadata proved an adequate linkage. If significant repeating questions were present in an instrument, an elaboration on this scheme would be desirable since in this scheme form, audio clips for repeating questions are themselves repeated in files of different names. The playing of audio clips is accomplished by the DLL routines simply by triggering a media player and passing the relevant ".wav" file for input.

[†] If written today, use of the Blaise™ Open Architecture rather than the API interface would be considered.

Likewise, use of a touch screen is accomplished by an operating system driver handshaking with the hardware device that translates screen touches into mouse click events. While significant methodological design should go into the use of these devices (primarily because the resolution of a human fingertip is very large), very little programming attention needs to be applied to their use.

3.3 Integrating Plug-in Blaise™ Components in a Windows Survey Management Environment

What is distinctive regarding Blaise™ use on the NHANES study is the usage of Blaise™ instruments as components of a much larger study framework. Of the many distinct NHANES applications, four of these are implemented using Blaise™ . All of these applications have been designed to be "plug-ins" to the Windows based Survey Management framework using a standard, well-defined interface.

The design of the instruments for NHANES required the use of topic oriented sections which could easily be re-used across other studies, and also allowed for accommodating change due to emerging health trends or refinement of the instruments.

Providing for Instrument Changes

A specific study goal in this cycle of NHANES was to support timely modification of study content to incorporate new research requirements as well as improvement of current content. The operational structure of the study, in one aspect, is well suited to absorbing such changes. As has been previously stated, approximately every three weeks a new round of data collection is started at a new field location. This is an opportune time to release content changes. Yet in another aspect, only one of three operational units is starting a new location at any one time. The other two units are in the middle of their local field cycle. There is no opportunity to easily release an instrumentation change to all interviewers simultaneously for the whole project. The NHANES data model allows for staged rollouts rather than complete rollouts of new data collection instruments.

Since small changes in Blaise™ instruments can generate incompatible data models, making complete rollouts on a fixed schedule much easier to manage. Supporting staged rollouts where multiple operating units are simultaneously generating data into incompatible Blaise™ data models is difficult. Native Blaise™ methods for unloading large portions of data from Blaise™ when dealing with different data models are limited.

Using Blaise instruments as plug-in components to the NHANES survey management architecture throughout the study data accommodated the high level of change requirement and operational schedule. As necessary to introduce new or modified content, new Blaise™ modules are "plugged into" the management system operations. Blaise™ instrument versions could be made that fulfilled this first behavior of plug-in components. They could accept regular preloaded data and run interview sessions. However, since changed instruments operated against different Blaise™ data models, their resulting, native Blaise™ data could not be readily combined. In addition, routines that had to extract critical operational data from completed interviews were potentially reaching into different storage structures each time they operated. To qualify as plug-in components, defined input and output from each component that was flexible to account for changing content was needed.

Data Flow Design Issues

As plug-in components, both advantages and disadvantages accrue in the Blaise™ applications. For instance, an extensive study management system is external to the Blaise™ instrumentation and alleviates the need for the instrumentation to do much more than report disposition status for case management aspects. However, the content of the NHANES questionnaires is closely linked to other external study components requiring immediate data extraction from these instruments. Further, these data are required for safety and protocol reasons in subsequent medical examinations so requirements of accuracy and timeliness are paramount. Thus, a requirement that a subset of interview data had to be accurately extracted in real-time and passed back to the management system for the study became a primary feature of the data flow design. The complicating factors were the closed native data architecture of Blaise™ databases and sustaining this operation through cycles of ongoing instrument modification. While a similar issue was faced in loading the Blaise™ instruments, the amount of preloaded data was very small and reasonably stable. Therefore, conventional Blaise™ methods of initializing data model fields through a simple Manipula™ script sufficed.

Once the subset of critical operations data was extracted into the management system, that system could handle this data further. Still, there remained thousands of general interview content data fields that also needed handling. The primary task for the operationally non-critical interview data is transmission to the central data aggregation location. This became a second data flow design issue. Again, a complicating factor was the anticipated level of ongoing instrument modifications. Aggregating all the field records in native Blaise™ database format was clearly not possible since multiple and incompatible data models would be in operation simultaneously.

Simply having multiple instrument data models presented another unavoidable data flow issue. At some point, these must be reconciled into a single, cohesive database. Methods were developed to handle the various data model versions and collapse them into a continually evolving single study database.

Methods used to successfully respond to these design issues are presented in the following section. These issues are not hardware or platform issues, rather they are data flow issues. The issue being how to get the correct data to the correct processing environment in the correct form, when it is needed.

These data flow issues all confront two current limitations of the Blaise™ system. Easy access to the native data storage is limited and Blaise™ data model structures have poor tolerance for change. Therefore, between the study management system and the Blaise™ instrument components these issues had to be resolved.

3.4 Methods Used for Managing Blaise™ Data and Models

The key methods used to address the critical data management and operational issues described above are presented in this section. These methods include:

- Extracting Blaise™ Components – A Fully Relational Method
- Capturing and Managing Blaise™ Instrument Data Sets – A BLOB Method
- Reconciling Blaise™ Data Model Changes Using Cameleon™
- Comparing Data Models to Identify Change

The techniques described in this section to transform data to SAS™ from Blaise™ are appropriate for instruments of moderate size and complexity. For extremely large and complex Blaise™ data models, other techniques should be applied. Solutions successfully developed and applied for large and complex instruments involve detailed recognition of Blaise™ metadata and extracting the Blaise™ data model to intermediate relational form based on this metadata.

Extracting Blaise™ Components – A Fully Relational Method

In general, native Blaise™ data extraction methods suffer from similar change intolerance as Blaise™ data models (since they are closely dependent). Assorted dumped output formats change with variations in the source data model. Also, Blaise™ tools (Manipula™) are generally oriented to producing fixed format output. While they can be programmed for some variation, the inherent standard fixed formatting is not flexible for accommodating unanticipated change. Therefore, a more flexible concept for extracting the large subsets of operational data from the questionnaire instruments in NHANES was needed.

The concept pursued for operational data extraction was the use of a fully relational data handling method for unloading actively needed data from Blaise™ instruments. This is not a new concept in general IT methods. Yet, it is not often applied to Blaise™ data since these data do not readily yield to a fully relational structure. Native Blaise™ structure is essentially horizontal with many fields of data on few rows. A fully relational structure is essentially vertical with few fields on many rows, thus reducing data redundancy. Therefore, reducing the columns and increasing the rows better leverages the manipulative potential of the data, offers greater flexibility of data combinations, and is more change tolerant.

The NHANES project uses a full-featured relational database (Sybase) for its overall project data repository. Therefore, moving Blaise™ data into such a structure was a necessity. The most obvious approach to this was taking inherent relationships in the Blaise™ instrument data and composing them as tables in the relational database. However, the rigidity in columns of relational tables, aggravated by version constraints of Blaise™ data models, and compounded by the rapid pace of instrumentation change proved to make this a labor intensive and error prone process.

What a fully relational method offers is that each discrete piece of data is handled as one data record. The main advantage of vertical structuring is overcoming row-based Blaise™ data model version constraints. With each data field represented as one row, the contents of each field can vary without impacting other fields represented on other rows. So instead of defining Blaise™ export data models of many fields where changes in one field impact other fields or the portability of the data model version in general, a single export data model of generic structure was used. This generic structure remains a fixed output construct, but its content granularity is so low that each data point becomes an independent unit.

Still, when this is done, a new data management problem appears. Once each field of data is disassociated from all others, identifying and managing the further use of these data must be addressed. The solution was to type each data field and create a new level of extract metadata within the study data management system. In practice, the assignment of type for each field amounted to assigning a unique numeric key and building a management table tracking all these numbers[‡]. An instance numbering scheme within field types was implemented so repeating instances of data could be numbered as the same field. This was not strictly necessary but reduced the proliferation of field numbers for common repeating data, increasing human comprehension of the metadata and making it marginally easier to handle a smaller metadata set.

One more necessary property of the single field per row description was a case identifier to associate the data field with its relevant case. Then the extraction data model was completed with a generic string field to hold the actual data value. All extracted data had to be represented in text format. Since most Blaise™ extract formats involve reduction of values to text, this is not a limitation unique to this extraction method. The resulting generic row extraction data model appears in Figure 3.3 (as rendered in Manipula™ code).

Figure 3.3, Simple Generic Data Model for Extraction Records

```
DATAMODEL UnloadRow
  FIELDS
     CaseID :      STRING[13]
     FieldNo :     INTEGER[4]
     InstNo  :     INTEGER[3]
```

[‡] If implementing this concept today, XML output for each field would be formed instead of using this field numbering scheme. XML is ideally suited to this purpose of representing data and metadata description in stand-alone form. XML tags could replace numeric keys in the management table.

```
            Resp    :       STRING
        ENDMODEL
```

With this data model in hand, the next step was to execute the actual extraction of desired fields from the Blaise™ collection data models. While the questionnaire instruments themselves could have output the extract data prior to terminating the interview, Manipula™ is a better-suited tool. Using a Manipula™ program separates this task into a discrete process. This makes version control more manageable and reduces the time required by the interviewing instrument to extract the necessary data.

The Manipula™ program to extract the data consists of a statement for each desired data field. The use of functions to organize the output of each unloaded row was beneficial. Each data value can be passed as a parameter to a function along with its field and instance identifiers. Accommodating different data types is problematic in the Manipula™ function calls. Therefore, a sequence of functions was developed, each handling a different data type. Examples of function code for integer and date/time data types appear in Figure 3.4.

Figure 3.4, Two Manipula™ Functions for Data Extraction

```
PROCEDURE NumOut
  PARAMETERS
    IMPORT P_Qno : INTEGER
    IMPORT P_INo : INTEGER
    IMPORT P_Ans : INTEGER
  INSTRUCTIONS
    if (P_Ans <> EMPTY) then

WriteOut(P_Qno,P_Ino,str(P_Ans))
    Endif
ENDPROCEDURE
```

```
PROCEDURE DateTimeOut
  PARAMETERS
    IMPORT P_Qno : INTEGER
    IMPORT P_INo  : INTEGER
    IMPORT P_dstr : STRING
    IMPORT P_tm   : TIMETYPE
  INSTRUCTIONS
    if (P_tm <> EMPTY) then
      Astr1 := P_dstr + ' ' + timetostr(P_tm)
      Writeout(P_Qno,P_Ino,Astr1)
    endif
ENDPROCEDURE
```

Each of these functions takes data from typed parameters and reformats it as text depending on the details of the necessary conversion. These data are then passed to a further common function to populate a row of output using the extract data model construct. In addition, a test is performed to eliminate plainly empty values from written output. Using these function calls, mainline Manipula™ programs can be created.

The handling of missing values is a weakness, prompted by parameter passing limitations in Manipula™ . The problem arises in passing specific data fields as generic parameters to Manipula™ functions. Apparently, Manipula™ disassociates the value from the field description necessary for it to subsequently recognize the value as missing in the function. The resolution was to identify where missing value content was required and code specific conditionals to test for these situations in the program. In addition, the Blaise™ native missing values were converted to a "DK" or "RF" string representation. Fortunately, in subsequent data handling, very few instances were found where required retention of missing values was significant. For most data fields, effectively equating explicit missing values with empty values and eliminating them from extraction was acceptable. If significant missing values occurs more frequently in the data, another coding approach is probably warranted.

Upon executing this application, the resulting output is a series of rows with one data value appearing on each row. Each row is identified with the project case identifier, a field identifier and the instance identifier. An example showing a subset of output is shown in Figure 3.5.

Figure 3.5, Raw Data Extraction[§]

```
2250300020102  12  111
2250300020102  50  125
2250300020102  50  211
2250300020102 184  1Programmer
2250300020102 197  1Greek
2250300020102 425  1allergies
2250300020102 750  1ALLEGRA
```

This example contains a diverse extraction of fields, an education attainment response (#12), code-all field with two instances (#50), occupation text (#184, Programmer), ethnicity specify (#197, Greek), medical condition (#425, allergies), and product associated with that condition (#750, Allegra®).

The prime advantage of this structure is not readily apparent from such a small example. However, considering that the current NHANES study extracts many hundreds of distinct data fields per instrument, with a three-week cycle of potential data model variation, the advantage becomes more apparent. When each extracted field is isolated from dependence on other fields, each can be handled per distinct requirement. When a field is empty or dropped from an instrument, it simply does not appear in the extraction. Conversely, new data fields simply begin to appear in the extraction at their release point. None of these occurrences have an effect on other fields or on the structure of the output files. This approach significantly reduces software maintenance resources required to accomplish the data extraction process as well as reduces error that may occur during the process.

The key to consuming the large number of distinct extracted records is developing an extract metadata description of these fields. This was accomplished by using the field number as a numeric key into a table of metadata. That table contains descriptive columns for subsequent data processing steps. For example, the desired data type and size for a field supports the conversion of text data into a relevant type in the relational database (e.g. fixed decimal accuracy or date/time stamp). Also, text field ids (Blaise™ question numbers), question text (populated separately from a Cameleon™ pass of the Blaise™ questionnaire data models), any specific storage table and column in the relational database, and further and special recoding designations are all associated in the metadata tables. An example of this meta-description appears in Table 3.6 (corresponding to the output example in Figure 3.5). This metadata is easily structured to meet varied requirements.

Table 3.6, Table Subset of Extraction Metadata Description

Field #	Quex.#	Quex. Text	Data type	Data size	Target table	Target column	Recode type
12	DMQ140	What is the highest grade or level of school …	Int	2	NH_Person	Person_grade	1
50	DMQ260	What ethnicity does SP consider (himself/herself) to be?	Int	2	NH_Person	Person_ethn	41
184	OCQ240	What kind of work was SP doing?	Text	180			100

[§] This figure contains mock data, not real study responses nor id. information.

197	DMQ260os	ENTER SPECIFY TEXT - OTHER RACE	Text	30			100
425	RXQ290	What is the main reason for which SP uses ^PrsMeddisp?	Text	60			100
750	RXQ240os	SELECTED MEDICATION NAME	Text	60			99

Importing the extracted rows into the database is simply a data import task. Since each record is handled once during this import task, this is also a good time to identify the data fields for critical or special processing. This is done by matching each field number from a read row to the metadata table. From the metadata any further actions are determined, such as converting the data from text format into that specified, storing critical data in any indicated location, and applying any recoding indicated. Then once loaded into this descriptive, fully relational storage, varied database views can be used to provide many specialized representations of extracted data. These views can be applied as input to further processing applications, support management processes and reports, quality control, or analytic activities.

Moderate effort is required in originating this metadata description on a large scale. Drilling into a large mass of it to make small ongoing changes also requires effort. However, the reward is a substantially independent field description that can readily absorb modifications, essentially eliminating the impact one field change may have on other data, and a data extraction/handling process that largely runs itself once configured. A further benefit of formalizing this description is a guarantee that this level of attention is applied to each and every data piece handled, improving the quality of the resulting dataset.

Capturing and Managing Blaise™ **Instrument Data Sets – A BLOB Method**

Most data collected in questionnaire instruments in NHANES are primarily used for analysis and research. As such, the main requirement for field systems is to completely and accurately capture the responses and deliver them for back-end aggregation. Each questionnaire potentially contains thousands of individual data fields. The majority of these responses are not immediately needed to support field operations. Therefore, extracting them in real-time, and thereby generating redundant storage of these items, makes no sense. Yet, they still require storage *en mass*.

Consideration was made to apply the previously discussed relational method of data extraction across all questionnaire items, not just those needed for further study management and safety/exclusion criterion. However, although the method is relatively efficient, running it on thousands of data elements at the conclusion of an interview was not possible. Further, removing the questionnaire data from native Blaise™ format immediately after the interview would bypass one of the great strengths of the Blaise™ system, the ability to edit data with the same rules as applied to collection. Therefore, keeping most of the interview data in Blaise™ format while in the field was advantageous.

Consideration had to be given on how the Blaise™ data sets should be organized. One approach would have been to aggregate all case records on each platform holding Blaise™ data. On an interviewer's computer, their entire assignment of cases, by instrument type would be in one Blaise™ database. On field office computers, all the cases for that location would be in one Blaise™ database. In addition, at the home office and client site this aggregation would be repeated for all worked cases. This would work if a single, stable data model could be used for each database. However, study change requirements precluded this assumption. In fact, multiple data models had to be accommodated in the field simultaneously.

Another design consideration was that the study already had a well-developed relational data base management system, including a replication architecture that provided near real-time data transmission of study data between the field offices, the home office, and NCHS. Incorporating Blaise™ data into this relational study backbone would join case management with case transmission. The first could leverage the second. Using tools of a relational database, a whole Blaise™ database in native format could be inserted as a file using a binary large object (BLOB) data type. If this were done per data model, per case, it solved the second requirement of having multiple data models in the field simultaneously. Having unloaded the operationally necessary data, the BLOBs could be pushed downstream in data flow where real-time response requirements are minimal. Back-end handling of multiple data model versions could be developed and applied when time for reconciling differences was available.

Implementing this approach proved to be straightforward, given appropriate file handling and database tools. The concept is that each Blaise™ case is stored singly in a Blaise™ database, combined with all its dependent files into a single file, and then inserted into a relational data record as a BLOB. If the case is again opened for work, this process is reversed to restore the Blaise™ instrument to its precise prior state. It was found that again preloading initial data is desirable to assure that any intervening changes to the preloaded data appear in the subsequent instrument execution.

The logical extension of this method suggests that among the Blaise™ database dependent files, the operative Blaise™ data model and meta-information files should also be stored. This creates actual version independent storage units for each case and instrument. However, a hefty file size price is paid for repeated storage of data model and meta information files. The scope of this size is megabytes. In contrast, the actual Blaise™ data files for the components in NHANES are typically measured in kilobytes. Therefore, only the actual Blaise™ data files were stored in the BLOBs and the data model version was tracked separately. Even with a relatively large number of data model versions, this remains a manageable number of files to manage. In addition, the number of data model versions active at any one time is limited.

This is an efficient and effective means of managing a repository of native Blaise™ data for subsequent use either in interviewing or post-processing. The ability to perform post-process editing and the capability to reuse Blaise™ instrument rules is preserved. The disadvantage of this method is that access to data items contained in the instrument data model is limited until the case is processed on the back end. It proves cumbersome without custom utilities to unwrap a case, accommodate its data model version, and access embedded data items prior to that point.

Reconciling Blaise™ Data Models, Post-processing using Cameleon™

Two major requirements were imposed on the post-processing of the current NHANES data. First, data from each field location would be delivered in formatted form within two weeks after the last data collection at that location. This dictated that on roughly a three week rotating cycle, some portion of Blaise™ interview data must be reconciled to eliminate released version variation and delivered for raw use. Second, initial delivery would occur as SAS™ datasets. Additional aggregation and reformatting into relational, analytically useful structures would occur subsequent to the SAS™ delivery.

Combining this fast turn-around time with changes in the instrumentation and it was obvious that a good method to deliver data rapidly was necessary. The conceptual approach was to reconcile data model variation by reading Blaise™ metadata and generating software to process the data, rather than attempting to originate this programming across all instrumentation on a rapid turn-around cycle. To read Blaise™ metadata, Cameleon™ was the utility available in the Blaise™ environment at that time. In practice, the method amounted to extracting the content data into ASCII output form with Manipula™ and simultaneously generating SAS™ application code using Cameleon™ scripts to process it.

The target deliverable consisted of SAS™ data sets organized by subject area content sections. Cameleon™ scripts were tailored to produce the various SAS™ and Manipula™ programs needed to combine all of the data from each data model version into a single Blaise™ database and then into

SAS™ data sets. To produce a set of scripts that need little modification as the data model changes, methods of making changes are guided by the need to maintain compatibility.

Rules were established to govern how changes are made to the data model. Rules state that if a question is to be deleted from the instrument, it is removed from the route, but it remains in the FIELDS section of the block. If the response categories for a question are going to be altered or additional responses are added, the new categories cannot reuse a response code that has been used at anytime during the life of that question in the current NHANES.

In addition, analysis software limitations determined item names of eight-characters in length. In this length, the content section, question number, array iteration and set iteration of any given field is described. The section and question number are designated in the first six character positions and the seventh position is preserved for array instances (e.g. family block repeated by persons). The eighth position designates set instances (e.g. repetitions of code-all-that-apply responses). No question is permitted to be an array itself, but is a field in an embedded array block to make it more easily detectable by the Cameleon™ scripts. Taking this approach to arrays produces a more relational model by having a record for each iteration of the loop

Individual Blaise™ data files are held for post-processing with each case contained in a BLOB in the main study relational database repository until batch post-processing is undertaken. Each case is then unloaded and unpacked so the enclosed Blaise™ database can be merged into a master Blaise™ database containing all of the interviews for that data model release. Then, the master Blaise™ file is converted to ASCII using a Manipula™ script. A SAS™ program then reads that raw ASCII file, re-codes the missing responses, re-orders the code-all-that-apply type questions and outputs a deliverable formatted ASCII file. A second SAS™ program reads the deliverable ASCII file, assigns variable labels, creates a format library and creates SAS™ and SAS™ Transport data sets. For quality assurance a list of the contents of the data set are provided to data preparation technicians for review along with a truncated dump of some interviews and frequencies of each data item.

Cameleon™ scripts are used to read the metadata and create the SAS™ software needed to read and manipulate the data. Three distinct scripts have been developed to handle the data. The scripts input the ASCII format data into SAS™, generate deliverable SAS™ and SAS™ Transport datasets, and convert the previous model's data into the current model. This last step effectively carries forward all previously processed data from each prior delivery stage into a single data model matching the current version release. This yields a single data set that constantly rolls forward, incorporating all data model changes.

Manipula™ setups perform the following processing: initialize the master Blaise™ database, add records to the master Blaise™ database, and create the ASCII output from the master Blaise™ database. At the time of a new data model release, each of these setups is copied to the processing area for the new model and then compiled against the new model. A small, automated procedure is used to parse the folder containing the cases. It executes the setup to initialize the master Blaise™ database and then processes the data package and adds records to the master as it encounters each case in the folder. The Manipula™ setup that creates the ASCII data file is run as soon as the last case has been added to the master Blaise™ database.

Comparing Data Models for Instrument Changes

As with most complex CAPI questionnaires involving changes over time eventually there comes a point when the details of changes between versions X and Y need to be identified. Even with the best version tracking, identifying and analyzing differences between data model versions at any two points in time can be laborious and error-prone.

While Blaise™ data models are optimized in many critical functional ways, they are not optimized for comparison. Further, with meta-information in a native product format, there is no easy way to figuratively open the hood on the data models and look at the internals.

Sven Sjodin outlined a solution to this limit in a previous IBUC paper[10]. Sjodin used the Cameleon™ utility to output the desired features of the meta information into external data records. After accomplishing this, analyzing two sets of records for variations deriving from instrument changes becomes possible. To facilitate the analysis, the output records were stored in a database. Then using this database as a meta information repository, broader and more flexible change analysis was possible.

The first issue encountered in implementing this process is the dearth of detailed documentation regarding Cameleon™ operation. A good description of programming syntax and a number of examples and sparse description of this utility program are provided in Blaise™ manuals and supplemental documentation sources. However, this documentation primarily describes how to run the utility and generate output in several standardized analytic formats. It does not offer functionally useful coverage of a number of peculiar aspects of Cameleon™ traversal of Blaise™ data models. Substantial experimentation with the provided examples can familiarize programmers with the nuances, but better and more complete description is desirable.

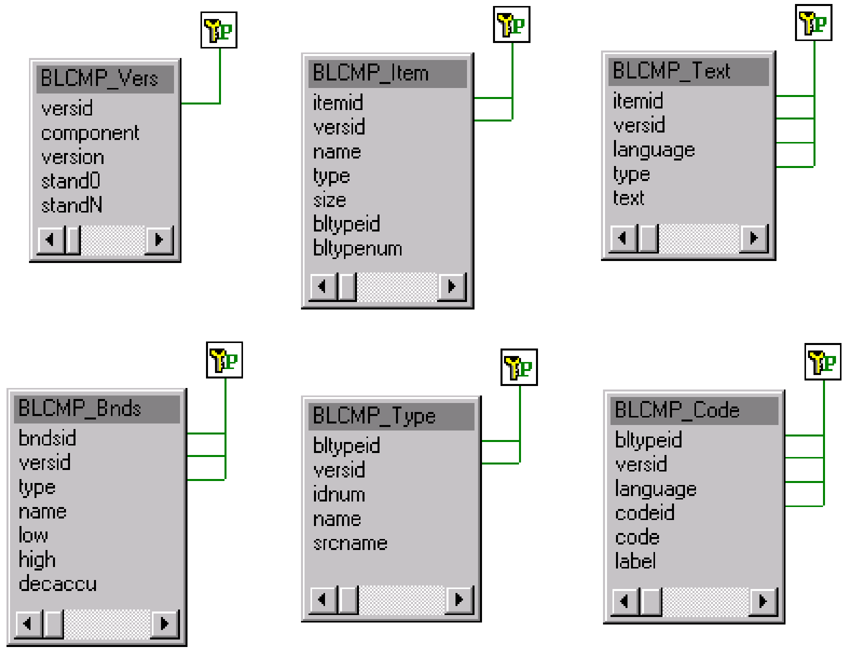
There are two potential disadvantages of this implementation. When the Open Blaise™ Architecture (OBA) is fully implemented the OBA methods will undoubtedly be an alternative approach to accessing similar meta information. Presumably, this could be extracted and saved as external data records as well. In addition, large instrument data models can overflow Cameleon™ 's memory capacity. The NHANES family component did this. Although it is of moderate size for a questionnaire, it has extensive block nesting, and produces a large data model. The Cameleon™ utility failed to process this instrument. While we can decompose the instrument into its various blocks and run each block separately, critical change detail is lost in the process. For example, the ability to compare the nested bounds of blocks appearing as arrays, or to compare the paths of nested blocks to identify some varieties of moved fields is essential to analyze all change possibilities in this instrument. In addition, this alternative is imprecise since it presupposes that source code is available for each version and that manipulations of two sets of source code are performed identically. Unfortunately, these limits appear when processing larger or more complex instruments, precisely where the benefits could be the greatest.

After determining what Cameleon™ could provide as output, six meta information entities were defined that were critical in distinguishing versions:

- Version tracking (component, version #, range of use)
- Item detail (name, data type, size, response type) *[for fields, blocks and types]*
- Question text (language, type, text)
- Item bounds/accuracy/range (name, low bound, high bound, decimal accuracy) *[for arrays, sets, ranges and decimal precision]*
- Response types (name, indirect name)
- Response codes (code, label, language)

Relational data tables were constructed to hold the data for each entity. While some relationships exist among these tables, it was easier to process the meta information without enforcement of constraints or foreign keys. This is because Cameleon™ outputs the data to be loaded in a somewhat indiscriminate order causing constraint violations upon insertion to the relational data base. Instead of reordering this output, it was easier to import the data and resolve dependency variation as part of the version comparison. A diagram of these tables appears in Figure 3.7.

Figure 3.7, Relational Tables of Cameleon™ **Meta Information Output**



A Cameleon™ program was produced to output meta information detail. A number of tagged records for each type of data loaded with related fields attached was sufficient. The records are mostly of fixed columns with a trailing open text last field where relevant. This format is a hybrid between the bundling of information from the Cameleon™ output and the target data structure for storage. These tagged records are:

```
FItm:0:  id(16) type(4) size(4) type#(6) type name(N)
FLbl:0:  id(16) label(N)
FPth:0:  id(16) path(N)
FTag:0: id(16) tag(N)
FTxt:lang: id(16) text(N)
Type:0: id(16) id2(16)
TypN:lang: id(16) code=label(N)
T###:lang: id(16) code=label(N)
Arry:0:   id(16) low:high
SetX:0: id(16) low:high
Rang:0: id(16) low:high
```

The Cameleon™ script then traversed the data model and produced these tagged output records. A data loading program to read the output tagged records, parse them, and store the contents in the database structure was then executed.

When the coding and database preparation was complete, the next step was to populate the tables. For each Blaise™ data model, the Cameleon™ program is executed against the meta information. This produces an output file of assorted rows. These rows become input records for the program above, which populates the individual tables. When completed for all data model versions of interest, a comprehensive database of Blaise™ meta information is ready for analysis. The available comparisons (in all languages of the instrumentation) include:

1. Blaise™ items, including blocks, fields, types, and their major attributes.
2. Text items, including question text, tags and labels, also path names.
3. Bounds for arrays and sets.
4. Ranges and decimal accuracy for numeric responses.
5. Type headers.
6. Code values and labels, for both types and individual fields.

Given this data set, identifying data model changes between versions involves simple comparisons. A series of SQL based reports, each targeting a different aspect of the data model were used.

From these SQL scripts, various reports can be constructed that organize and display the output. An example report is provided in Figure 3.8.

Figure 3.8, Example of Version Comparison Report

Modified Items Report Stand: 112 to 122 (Versions: Stand 112b to Year2 122)

Item	Size	Dimen.	Decimal	Version
Text Change (English)				
ACQ0101				Stand 112b
What language(s) ^sdoyouSP usually speak at home? CODE ALL THAT APPLY				
ACQ0101				Year2 122
Now, I'm going to ask you about language use. What language(s) ^sdoyouSP usually speak at home? CODE ALL THAT APPLY				
New Item				
SMQ077	1			Year2 118
How soon after you wake up do you smoke? Would you say ...				
Dim. Change				
DMQ2501		1 : 8		Stand 112b
DMQ2501		1 : 9		Year2 122
Dropped Item				
DMQ2509	2			Stand 112b
HAND CARD DMQ3HELP AVAILABLE Please give me the number of the group that represents ^syour Hispanic origin or ancestry. Please select 1 or more of these categories. SELECT 1 OR MORE				
Size Change				
DSQ050	60			Stand 112b
DSQ050	120			Year2 122

4. Conclusion

The NHANES is a large, complex, nationally important survey resulting in the collection and dissemination of a unique data set to the health research community. The interviewing and data management requirements presented by the project are challenging. Blaise™ software used on NHANES must: operate effectively on varied Windows based platforms; interface with an extensive distributed survey management system; accommodate significant, designed requirements for change; access and integrate metadata; capture and manipulate collected data; and convert Blaise™ collected data to relational and SAS™ representation for delivery and use by the public health community. Blaise™ and its associated tools, Cameleon™ and Manipula™ have provided the foundation to implement solutions to successfully meet the challenging requirements of this project.

References

1 Looker AC, Dallman PR, Carroll, MD, Gunter EW, Johnson CL. *Prevalence of Iron Deficiency in the United States*: JAMA. 1997; 277:973-976.

2 Looker AC, Orwoll ES, Johnston CC, Lindsay RL, Wahner HW, Dunn WL, Calvo MS, Harris TB, Heyse SP. *Prevalence of low femoral bone density in older U.S. Adults from NHANES III*. J Bone Mineral Research, 12:11, 1761-1768, 1997.

3 Ogden C, Troiano R, Briefel R, Kuczmarski R, Flegal K, Johnson CL. *Prevalence of overweight among preschool children in the United States, 1971-1994*: Pediatrics, 99(4). 1997.

4 Ernst N, Sempos C, Briefel R, and Clark MB. *Consistency between US dietary fat intake and serum total cholesterol concentration: The NHANES Surveys*. Am J Clin Nutr. 66(suppl) 956S-972S. 1997.

5 CDC Growthcharts homepage: http://www.cdc.gov/growthcharts/

6 Blaise™ homepage in U.S.: http://www.westat.com/blaise/

7 Powerbuilder homepage: http://my.sybase.com/detail?id=47737

8 Shepherd J, Hill D, Bristor J, Montalvan P*, Converting an Ongoing Health Study to CAPI: Findings from NHANES III*. Conference on Health Survey Research Methods, Denver, Colorado. 1995.

9 Devlin B. Data Warehouse from Architecture to Implementation. Addison-Wesley Longman, Inc., 1997, pp 41-62.

10 Sjodin S, *Whatever Happened to our Data Model?,* 6th International Blaise™ User Group Conference, Kinsale, Ireland, May 2000