Programming Techniques for Complex Surveys in Blaise

Ventrese Stanford, US Census Bureau TMO Authoring Staff
David Altvater, US Census Bureau TMO Authoring Staff
Curtis Ziesing, US Census Bureau TMO Authoring Staff

Software organizations are constantly seeking strategies to improve their ability to develop and deliver high quality software in a timely and efficient manner. The U.S. Census Bureau's Technology Management Office (TMO) Authoring Staff is implementing the Capability Maturity Model (CMM)[1]as a strategy to improve their software development process. The CMM identifies five levels of software process maturity, from the initial chaotic stages of development (Level 1) to an optimized, mature process (characterized by Level 2 through Level 5). This paper focuses on some of the software tools the Census Bureau's Authoring Staff developed as part of the strategy to support the Level 2 CMM methodology. The tools are used as a strategy to establish continuous process improvement. By establishing both the software techniques and the methodology to effectively manage those tools in future projects, we support the Level 2 CMM objective - Repeatability. Effective and efficient programming techniques are identified and documented and will serve, with other process improvement tools, as the basis for developing reference materials for new projects. These successful techniques are proposed as a standard across Blaise instruments developed by the Census Bureau.

The Census Bureau recently added the Blaise 4 Windows Software to their box of programming tools for automating surveys. In converting to this language from the DOS based CASES language that has long been the standard in the Census Bureau, many prototypes were developed to test complex data collection requirements in the Blaise environment. This presentation will present some of these complex prototypes and other challenging requirements that will serve as the basis for the Census Bureau's computer assisted interview (CAI) standards through a discussion of some of the screen and internal design strategies. These designs support the practice of repeating successful practices in future automation survey development projects. And finally, the discussion concludes with some insight from some of the lessons learned and suggestions for future Blaise software enhancements.

Screen Design

The objective in the screen standard design is to provide an interviewing environment that is meaningful, familiar over time and responsive to the multiple needs of interviewers during data collection. This objective has resulted in standardizing features ranging from the use of various function keys to the color and size of screen text. Every attempt is made to develop an interviewing environment that is efficient with a look and feel that is similar across Blaise surveys. Also considered in the Blaise design are CASES design elements. As the Census Bureau's makes the DOS to Windows conversion in surveys, some design elements have been carried forward from the DOS based CASES surveys to minimize interviewer confusion. For example, when developing standard Function Key number assignments in Blaise, whenever possible the assignments commonly used in CASES surveys were adopted.

Starting with the basic "out of the box" Blaise default screen displays, described below are some standards that have been established for the text, color, interviewer instructions, navigation guides, reference information and errors. Implicit in the design is the effort to clearly display as much information as needed on one screen.

[1] *The Capabiltiy Maturity Model : Guidelines for Improving the Software Process*, Carnegie Mellon University, Software Engineering Institute, 2000,Addison-Wesley Longman, Inc. Boston, Ma.

Book and Diamond Symbols

Since "a picture is worth a thousand words," symbols are used wherever possible as instructions to the interviewer. In Figure 1 below, the book symbol indicates to the interviewer that they should display page 13 of a bounded information booklet for the respondent to review. The diamond symbol indicates that the following text is not to be read but is an instruction to the interviewer. The book and diamond are created using a True Type Wingdings font. The character display is set to char (38) for the book and char (115) for the diamond. The following code shows the assignment:

```
AuxBook      := '@Z'+Char(38)+'@Z'
AuxDiamond := '@Z'+Char(115)+'@Z'
```

Auxbook and AuxDiamonds are string auxfields. The @Z is a user defined font in the modelib that has been assigned the True TypeWingdings font. When the book or diamond is needed to display in the question text area, the parameter is passed to the block and referenced as a variable in the question text. The following code shows the call to the block with the parameter passing and the actual field question text setup that displays the symbols.

```
Section1(AuxBook, AuxDiamond)

Block Section1 "Called Block"
  Parameters Import AuxBook, AuxDiamond : String
  Fields
    Field1 "^AuxBook @/@/ ^AuxDiamond Question text"
Endblock
```

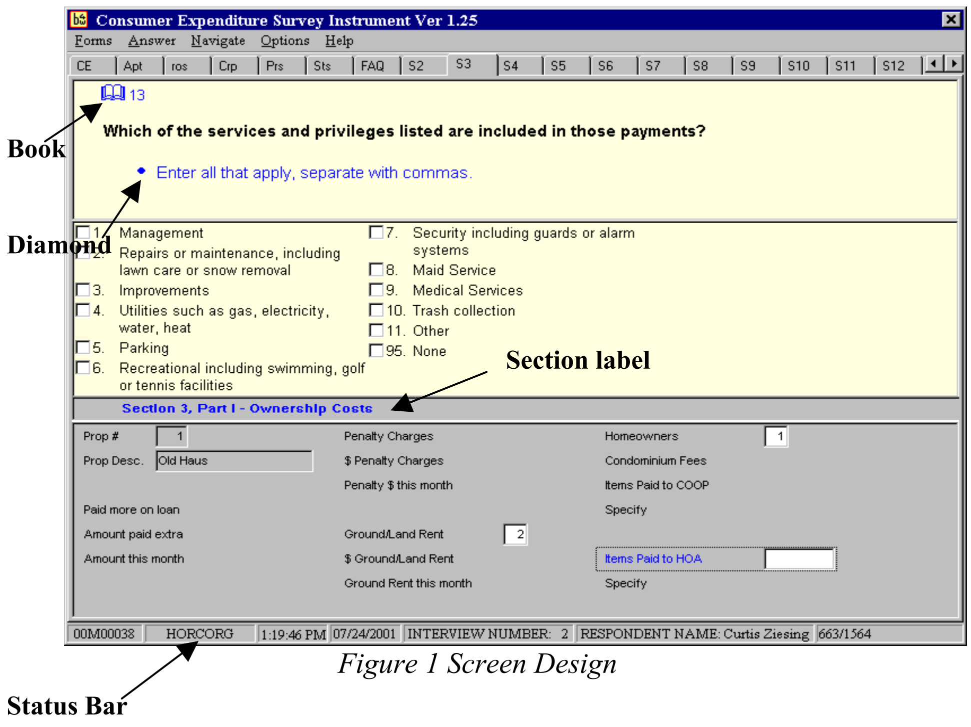
Windows uses this type of visual identifier by creating the icon and Blaise displays the icon using the ^ command. The @/@/ indicates to Blaise to advance two lines down in the screen. The result of this technique is a small picture the user can clearly and quickly identify as specific instructions for the current question.

Status and Speed Bars

The standard screen display information available has been increased to give the maximum space real estate. For example, the Blaise Speed Bar was eliminated. The speed bar was removed because the Census Bureau laptop systems are focussed on keyboard data entry. The speed bar is a GUI feature that relies on the use of the mouse. The same graphic user interface (GUI) functionality is offered elsewhere on the keyboard. This elimination resulted in some space savings on the display screen. Also, the flexibility of Blaise enabled the development of standard information to display on the Status Bar, such as case identification number, date, interview number, respondent name, block page number, and field name for reporting problems.

Font Type, Color and Size

Font type color and size are used consistently to guide the interviewer in reading text. The Census Bureau has established GUI screen standards of font type size and color. For surveys developed in a graphic environment the Bold Times Roman font size12 text indicates to the interviewer information that is read to the respondent, blue text are instructions to the interviewer, and gray text are optional text that are read to the respondent only when necessary. Figure 1 below illustrates this standard. Grey text usually occurs when collecting data for repetitive lists. The text is displayed in bold black the first time it is read. After the respondent's first response, the text is displayed in gray indicating to the interviewer that the leading phrase in the question is read only if necessary for continuity and clarity.


Consumer Expenditure Survey Instrument Ver 1.25
Forms Answer Navigate Options Help
CE Apt ros Crp Prs Sts FAQ S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12
Book
Diamond
Section label
Status Bar

13

Which of the services and privileges listed are included in those payments?

• Enter all that apply, separate with commas.

☐ 1. Management	☐ 7. Security including guards or alarm systems
☐ 2. Repairs or maintenance, including lawn care or snow removal	☐ 8. Maid Service
☐ 3. Improvements	☐ 9. Medical Services
☐ 4. Utilities such as gas, electricity, water, heat	☐ 10. Trash collection
☐ 5. Parking	☐ 11. Other
☐ 6. Recreational including swimming, golf or tennis facilities	☐ 95. None

Section 3, Part I - Ownership Costs

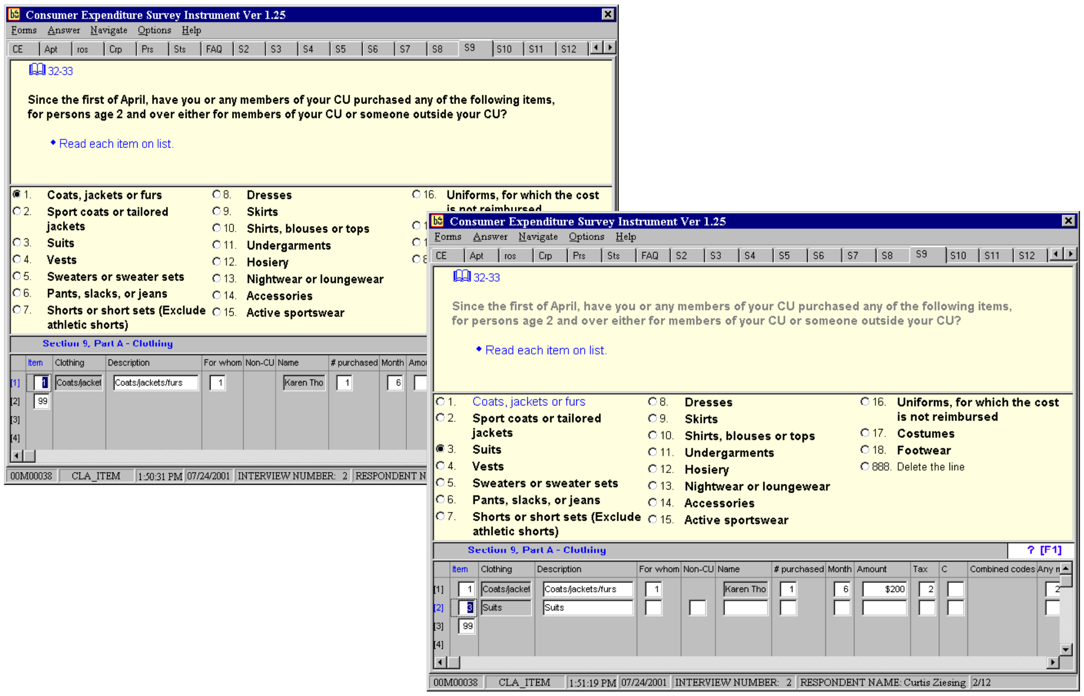
Prop #	1	Penalty Charges		Homeowners	1
Prop Desc.	Old Haus	$ Penalty Charges		Condominium Fees	
		Penalty $ this month		Items Paid to COOP	
Paid more on loan				Specify	
Amount paid extra		Ground/Land Rent	2		
Amount this month		$ Ground/Land Rent		Items Paid to HOA	
		Ground Rent this month		Specify	

00M00038 | HORCORG | 1:19:46 PM | 07/24/2001 | INTERVIEW NUMBER: 2 | RESPONDENT NAME: Curtis Ziesing | 663/1564

Figure 1 Screen Design

The code below, illustrates how the black text changed to dark gray in Figure 2. In this example, if the first item in the laundry list has been answered yes and there are more questions, then the flag LTGrey which is an auxfield is assigned the value yes. When the LTGrey flag equals yes and there are more questions then the auxfield DKGray is populated with the user defined font @K. The @K has the value for the font size, the color gray and its text type.

```
{Ask the screening question and set the flag}
   Item
     IF Item = NoMore Then
       LTGrey := No
     ELSE
       LTGrey := Yes
     Endif

{If the flag is set and it is not the last question then make the text gray}
IF LTGrey = Yes AND NOT ITEM = NoMore THEN
      DKGray := '@K'
    ELSE
      DKGray := ''
    ENDIF
{Assign the appropriate color in the question text}
    Question :=  DKGray + 'Since the first of  last month  have you paid for -' + DKGray +
                    '@|'  + ' Read each item on list.'

 {Ask the question}
    Question.ask
```


Consumer Expenditure Survey Instrument Ver 1.25
Forms Answer Navigate Options Help
CE Apt ros Crp Prs Sts FAQ S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12

32-33

Since the first of April, have you or any members of your CU purchased any of the following items, for persons age 2 and over either for members of your CU or someone outside your CU?

- Read each item on list.

[✓ 1.] Coats, jackets or furs	8. Dresses	16. Uniforms, for which the cost is not reimbursed
2. Sport coats or tailored jackets	9. Skirts	
3. Suits	10. Shirts, blouses or tops	
4. Vests	11. Undergarments	
5. Sweaters or sweater sets	12. Hosiery	
6. Pants, slacks, or jeans	13. Nightwear or loungewear	
7. Shorts or short sets (Exclude athletic shorts)	14. Accessories	
	15. Active sportswear	

Section 9, Part A - Clothing

	Item	Clothing	Description	For whom	Non-CU	Name	# purchased	Month	Amou
[1]	1	Coats/jacket	Coats/jackets/furs	1		Karen Tho	1	6	
[2]	99								
[3]									
[4]									

00M00038 CLA_ITEM 1:50:31 PM 07/24/2001 INTERVIEW NUMBER: 2 RESPONDENT N

Consumer Expenditure Survey Instrument Ver 1.25
Forms Answer Navigate Options Help
CE Apt ros Crp Prs Sts FAQ S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12

32-33

Since the first of April, have you or any members of your CU purchased any of the following items, for persons age 2 and over either for members of your CU or someone outside your CU?

- Read each item on list.

1. Coats, jackets or furs	8. Dresses	16. Uniforms, for which the cost is not reimbursed
2. Sport coats or tailored jackets	9. Skirts	17. Costumes
[✓ 3.] Suits	10. Shirts, blouses or tops	18. Footwear
4. Vests	11. Undergarments	888. Delete the line
5. Sweaters or sweater sets	12. Hosiery	
6. Pants, slacks, or jeans	13. Nightwear or loungewear	
7. Shorts or short sets (Exclude athletic shorts)	14. Accessories	
	15. Active sportswear	

Section 9, Part A - Clothing ? [F1]

	Item	Clothing	Description	For whom	Non-CU	Name	# purchased	Month	Amount	Tax	C	Combined codes	Any n
[1]	1	Coats/jacket	Coats/jackets/furs	1		Karen Tho	1	6	$200	2			2
[2]	3	Suits	Suits										
[3]	99												
[4]													

00M00038 CLA_ITEM 1:51:19 PM 07/24/2001 INTERVIEW NUMBER: 2 RESPONDENT NAME: Curtis Ziesing 2/12

Figure 2 - Laundry list with the Text Bold and Then Gray

Section Labels

The Consumer Expenditure Survey (CE) has 23 major sections with questions on topics such as housing, clothing, and cars. Many of these sections contain subparts that focus on questions in a very specific area such as the housing section having subparts for mortgages and home equity loans. Blaise enables the interviewer to skip around to various questions during the interview. This random navigation results in the need for the screen to have distinct and clear identification of each section and part. This specification was addressed with the use of bitmaps as shown in Figure 3. These bitmaps were created using Microsoft Windows Paint although any software could be used. The bitmaps were then integrated at the field level in Blaise. For questions that have links to specific help screens, a second bitmap label was created adding the ?[F1] symbol. The ?[F1] symbol on the right of the screen informs the interviewer that the same information is also available on a help screen for them to follow in support of a particular question by simply pressing the F1 function key. Displaying these symbols result in an efficient use of the limited space available on the screen.

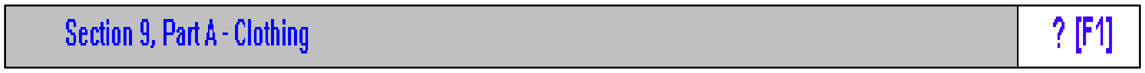

Section 9, Part A - Clothing ? [F1]

Figure 3 – Example of an Individual Section Label Bitmap

Code at the Field Level with the Bitmap Image

<u>Fields</u>
ITEM “Since the first of April, have you or any members of your CU purchased any of the following

Items for persons age 2 and over either for members of your CU or someone outside your CU?" "**image(info09aH.bmp, TOP=345, LEFT=0)"** : TSec9aItems

The code above shows the bitmap link including its location on the screen for the field ITEM, it is 345 points down from the top of the screen and starting 0 points from the left of the screen. The code was standardized in the illustration for placement of this bitmap by assigning the value to an auxfield. This shortcut enables easy revision for all fields in the block sharing the same screen layout.

Fields

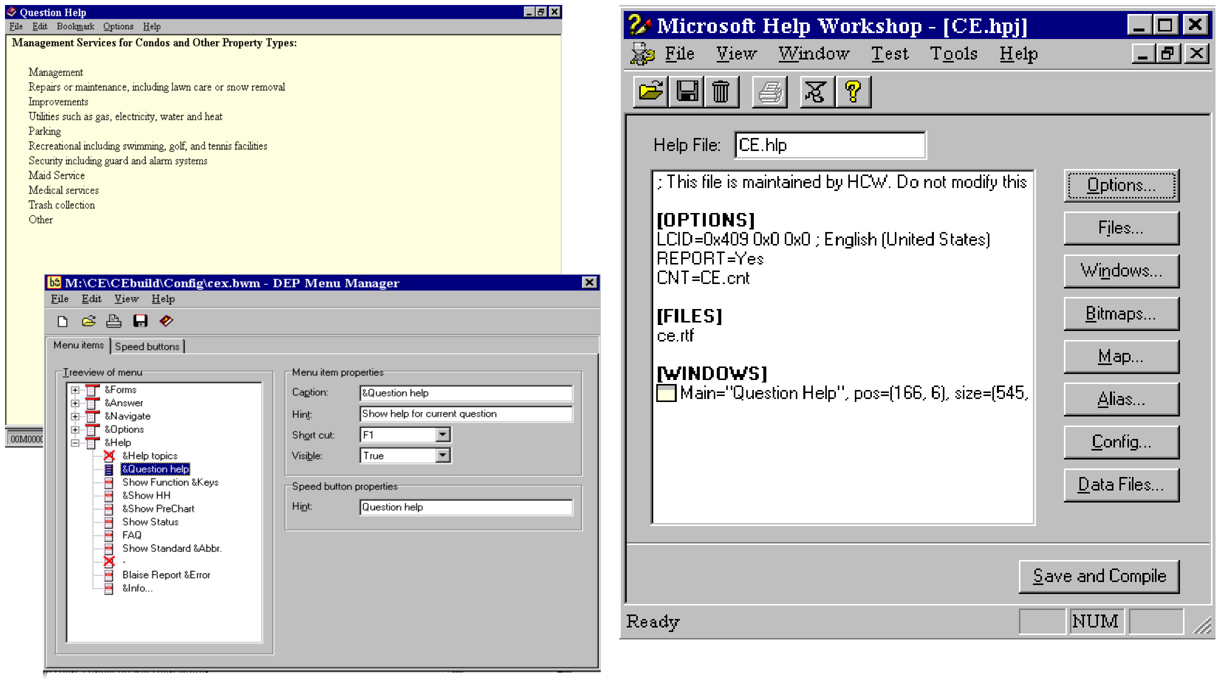
ITEM "Since the first of April, have you or any members of your CU purchased any of the following Items for persons age 2 and over either for members of your CU or someone outside your CU?" **"^TEMPIMAGE"**: TSec9aItems

Rules

TEMPIMAGE:= **'image(info09aH.bmp, TOP=345, LEFT=0)'**

Help Files

As previously described, the ?[F1] symbol indicates that a help screen is available. This screen is linked to the F1 function key. The help file is created as an external file linked in Blaise by using the DEP menu manager tool. The help file in this example was created using Microsoft Help Workshop but it could be developed in any help software program. The text can be written in any editor that supports rtf. Once the rtf file is created, it is compiled in a help format. The compiled file must have a .hlp extension for the Blaise link through the DEP menu manager as shown in Figure 4.



Management Services for Condos and Other Property Types:

Management
Repairs or maintenance, including lawn care or snow removal
Improvements
Utilities such as gas, electricity, water and heat
Parking
Recreational including swimming, golf, and tennis facilities
Security including guard and alarm systems
Maid Service
Medical services
Trash collection
Other

Caption: &Question help
Hint: Show help for current question
Short cut: F1
Visible: True

Hint: Question help

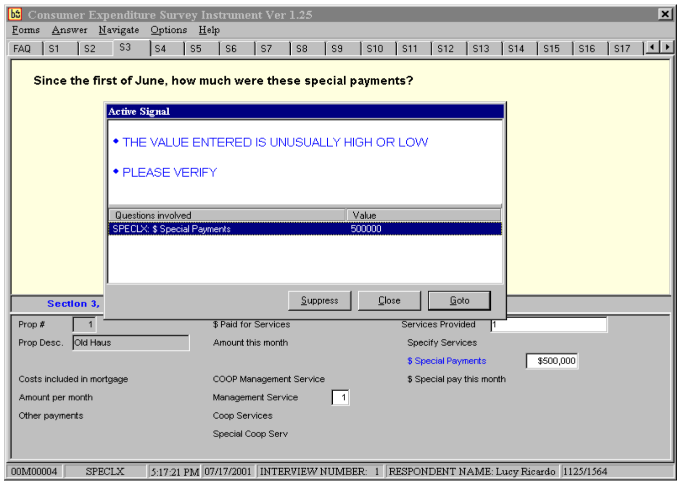
Help File: CE.hlp

```
; This file is maintained by HCW. Do not modify this

[OPTIONS]
LCID=0x409 0x0 0x0 ; English (United States)
REPORT=Yes
CNT=CE.cnt

[FILES]
ce.rtf

[WINDOWS]
Main="Question Help", pos=(166, 6), size=(545,
```

Figure 4 – Help Files

Error messages

The specifications for the CE survey required numerous edit checks for valid data ranges. Identified were six common error messages used in the checks. These messages were standardized and declared at the data model level. The messages are passed as parameters into various section blocks as needed. This method enabled an easy establishment of a standard in appearance and text across all sections in the survey that can be easily edited. The consistency enables interviewers to quickly interpret the problem and address its resolution.


Consumer Expenditure Survey Instrument Ver 1.25
Forms Answer Navigate Options Help
FAQ S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13 S14 S15 S16 S17
Active Signal
Suppress
Close
Goto
Section 3,

Since the first of June, how much were these special payments?

• THE VALUE ENTERED IS UNUSUALLY HIGH OR LOW

• PLEASE VERIFY

Questions involved	Value
SPECLX: $ Special Payments	500000

Prop #	1	$ Paid for Services		Services Provided	1
Prop Desc.	Old Haus	Amount this month		Specify Services	
				$ Special Payments	$500,000
Costs included in mortgage		COOP Management Service		$ Special pay this month	
Amount per month		Management Service	1		
Other payments		Coop Services			
		Special Coop Serv			

00M00004 | SPECLX | 5:17:21 PM | 07/17/2001 | INTERVIEW NUMBER: 1 | RESPONDENT NAME: Lucy Ricardo | 1125/1564

Figure 5

The code below is an illustration of how the error messages can be standardized at the datamodel level. The error messages are assigned to an auxfield at the datamodel and then passed into various blocks requiring theses specific types of messages. The message is displayed in the block upon entering an error. This technique enables easy editing of the messages and a standard continuity throughout the instrument.

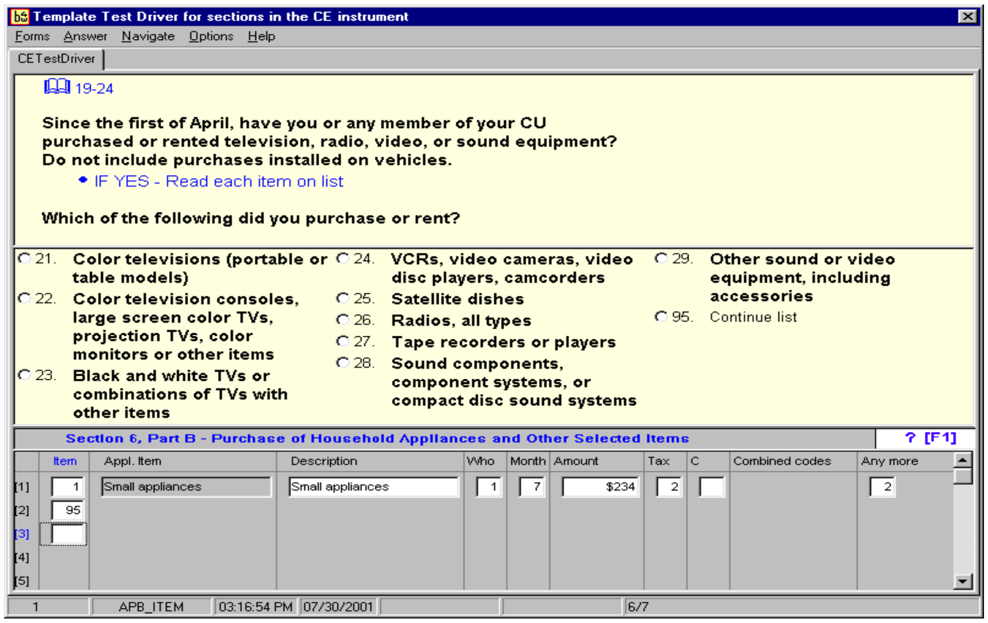
```
Datamodel
  Err1 := Diamond + 'THE VALUE ENTERED IS UNUSUALLY HIGH OR LOW' +
           Diamond + 'PLEASE VERIFY'
  Err2 := Diamond + '100% WAS ENTERED' +
           Diamond + 'PLEASE VERIFY'
  Err3 := Diamond + 'MONTH ENTERED IS NOT IN REFERENCE PERIOD' +
           Diamond + 'PLEASE VERIFY'
  Err4 := Diamond +'$0 WAS ENTERED FOR EXPENSE' +
           Diamond + ' PLEASE VERIFY'

   Section03 (Book, Diamond, Err1, Err4)
  Block BSection3
 Parameters
   Import
    Book, Diamond, Err1, Err4 : STRING
 Rules
  SPECIALX
   IF SPECIALX = RESPONSE THEN
     SIGNAL
       SPECIALX >= 8 AND SPECIALX <= 22400 "^Err1"
   ENDIF
```

Internal Design

The focus of the internal design for this paper is developing modules of code or methods of data collection within the software instrument which are reusable either within the same instrument or by other instruments, a repeatable methodology. Surveys in the TMO Authoring branch are designed considering the requirement of repeatable code that can be used throughout the instrument or in other instruments. Several examples of these designs are presented such as the laundry list in the Consumer Expenditures Survey, the open grid and the current status block developed in the American Community Survey's Telephone Follow-up project. This section concludes with some internal design strategies to improve instrument performance.

Laundry Lists

The 'laundry list prototype' was developed in conjunction with Mark Pierzchala of Westat. It is a set of modules used to collect expenditure data about a range of grouped and similar expenses on selected items. The structure of this prototype provides standard, reusable modules of code that are easily revised. These modules are used throughout the Consumer Expenditures Survey instrument with slight modifications for each instance. An advantage of the laundry list prototype is that the interviewer is presented with the same basic screen appearance while collecting different types of data. This feature enhances continuity and speed in the interview. In designing this prototype, the basic premise was to have a survey-programming concept that would allow the collection of any type of data even though it is segmented into specific parts. Examples of these parts are vehicles, medical payments, education expenses, clothing, and appliances. Each part is further segmented into various groups to collect more specific data. For example, Figure 6 shows Section 6, Part B of the appliance laundry list in the CE Survey. To distinguish between parts, a code of "95" is used as a separator. The first group within Part B is for small appliances. The next group is for larger appliances. The final group is for general equipment. When all similar items are completed the "95" is entered to indicate that the group is complete. The interviewer can then begin entering data for the next group. Once the data is collected for the final group, the interviewer enters a "99" to indicate the completion of the part.


Template Test Driver for sections in the CE instrument
Forms Answer Navigate Options Help
CETestDriver

19-24

Since the first of April, have you or any member of your CU purchased or rented television, radio, video, or sound equipment? Do not include purchases installed on vehicles.

- IF YES - Read each item on list

Which of the following did you purchase or rent?

- 21. **Color televisions (portable or table models)**
- 22. **Color television consoles, large screen color TVs, projection TVs, color monitors or other items**
- 23. **Black and white TVs or combinations of TVs with other items**
- 24. **VCRs, video cameras, video disc players, camcorders**
- 25. **Satellite dishes**
- 26. **Radios, all types**
- 27. **Tape recorders or players**
- 28. **Sound components, component systems, or compact disc sound systems**
- 29. **Other sound or video equipment, including accessories**
- 95. Continue list

Section 6, Part B - Purchase of Household Appliances and Other Selected Items ? [F1]

	Item	Appl. Item	Description	Who	Month	Amount	Tax	C	Combined codes	Any more
[1]	1	Small appliances	Small appliances	1	7	$234	2			2
[2]	95									
[3]										
[4]										
[5]										

1 APB_ITEM 03:16:54 PM 07/30/2001 6/7

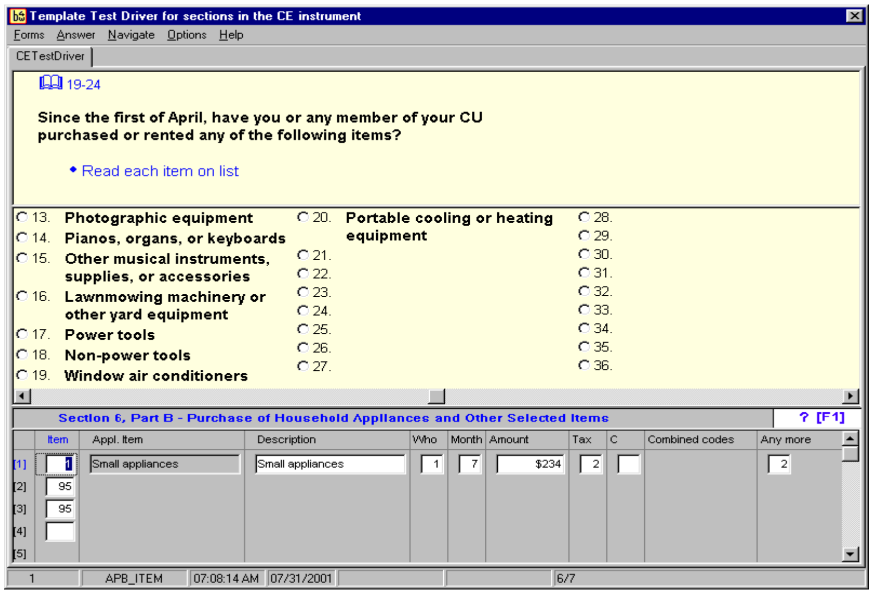
Figure 6 - Laundry List

In terms of the internal design of a laundry list, a generic shareable table layout was used. Collecting data in a table allows the user to visually access information as it is entered into the survey form. This method enabled the table to be 'plugged' into any section of the instrument that asked questions in a laundry list style. The table block for this instrument is called 'TTable'. By creating one generic shareable TTable, as modifications are required their results are reflected across all sections that use the laundry list. The technical side to the laundry list begins with a setup of the table itself. A field of type table would be defined. Within the definition of the table, there is a block of data specifically called BRow. Following that is the generic TTable code that has a field Row defined as type Brow as in the following snipit of code:

```
Datamodel Example
 USES LLMetaData
 Table BTableA {Blaise setup for rows and column input}
  Block Brow {each row would have many fields and error trapping}
   ...  {Parameters and other code would be here}
   EXTERNALS LMD : LLMetaData
   ...  {Other code would be here for locals and such}
   Type
    TSec6aItems = (Item1                (1)  "^TempItem[1]",
                   ...  {Additional selectable items would be here}
                   Item45               (45)  "^TempItem[45]"
                   NoMore95             (95) "^TEMP95",
                   NoMore99             (99) "^TEMP99",
                   DeleteLine           (888) "^DeleteLine")
   Fields
    Item : TSec6aItems
    ...  {Many other fields defined here}

 Rules
    IF LMD.SEARCH (SecNum, SecLetter) THEN
      LMD.READ
   ENDIF
    ...  {A lot of rules code would be here}
  Endblock {Brow}

  Fields
   Row : ARRAY [1..100] OF Brow
 Endtable {Ttablea}

 Fields
 TableA : BTableA
Endmodel {Example}
```

Within the block BRow there are some specific codes for the laundry list that are generic for all sections. This code controls the selection of "95" or "99". It also controls the change of an entered item by deleting the associated data if an item code were changed. There are many other error trapping routines to ensure the best and most accurate data is collected.

Another concept developed in conjunction with Mark Pierzchala of Westat is the use of metadata. This contains the text for question fills and selectable items and range edits for the CE instrument. It is a form of dependent data. The external metadata was designed in Blaise as a table. Each row in the table represents a section and its associated parts. For example, Section 6 and its two parts are identified by two rows in the metadata table. Once the external file was built, in the Blaise instrument at the data model level a USES statement establishes the metadata file for the interview. Within the block where the metadata file is used, the EXTERNALS clause is coded and then the actual code to call the file are the SEARCH and READ methods. Because this is general across all sections that use the laundry list, a generic section can be developed in a short amount of time, then modifications to cater to the actual section can be made. This allows focus to be placed on the data being collected and not so much on the method of collection. This technique also facilitates training, both programming and interviewing. Once the laundry list methodology is grasped it is leveraged throughout the instrument. The metadata sets the number of selectable items. So if there were 18 items listed in the metadata then only the first 18 items are filled. The challenge in this technique is to control the display of desirable items on the screen where there is less than the maximum in the array. Section 6, Part B has less than 45 items which results in displaying the empty array positions with selectable numbers and scrolling as displayed in Figure 7. To resolve this problem, the Blaise hide empty option was used, but the empty options could still be selected. An additional error trapping routine was developed to disallow the selection of empty items.


Template Test Driver for sections in the CE instrument
Forms Answer Navigate Options Help
CETestDriver

19-24

Since the first of April, have you or any member of your CU purchased or rented any of the following items?

◆ Read each item on list

○ 13. **Photographic equipment**	○ 20. **Portable cooling or heating equipment**	○ 28.
○ 14. **Pianos, organs, or keyboards**	○ 21.	○ 29.
○ 15. **Other musical instruments, supplies, or accessories**	○ 22.	○ 30.
○ 16. **Lawnmowing machinery or other yard equipment**	○ 23.	○ 31.
○ 17. **Power tools**	○ 24.	○ 32.
○ 18. **Non-power tools**	○ 25.	○ 33.
○ 19. **Window air conditioners**	○ 26.	○ 34.
	○ 27.	○ 35.
		○ 36.

Section 6, Part B - Purchase of Household Appliances and Other Selected Items ? [F1]

	Item	Appl. Item	Description	Who	Month	Amount	Tax	C	Combined codes	Any more
[1]	1	Small appliances	Small appliances	1	7	$234	2			2
[2]	95									
[3]	95									
[4]										
[5]										

1 APB_ITEM 07:08:14 AM 07/31/2001 6/7

Figure 7 – Hide Empties Not Enabled

Once the item has been selected, the text fields in the metadata are transferred to the text fills in the instrument. If the item changes, the question text will change to fit the current item. Each section and part can input as many as six text strings that can be used as fills in the instrument. This can be expanded and it shows the versatility of Blaise. Each time an item is selected the data from these string fields in the metadata are transferred to the instrument fields and displayed. The text fills are not stored in the code, or hard coded therefore text can be modified independently after the instrument has been prepared. If one of the fields in the row requires a specific edit range based on the item selected then the metadata will provide the high and low values for the edit check. For each item code, a different edit range is established through the use of the metadata file. Instead of having a long case statement, coding is reduced to using only one edit check and an error routine for any item code selected. A generic maximum number of 100 rows are used because each group needs to limit the number of rows collected. A variable called maxrows represents the limit for the number of rows collected for any given type. For some groups such as vehicles, a maximum of 25 items may be sufficient for collecting data; however for clothing, an entire household may be require a higher maximum. The maximum number of rows collected using the TTable needs to be limited to preserve instrument performance.

Designing a screen that displays all data for each part on the screen simultaneously can be very challenging when there are a large number of items. The amount of time to develop these visual aesthetics needs to be considered when planning project time lines. The result of these visual aesthetics is a smooth flowing instrument that conveys information to the interviewer in the most convenient manner and enhances efficiency. Font colors were also standardized in the laundry list prototype to enhance the flow of the interview. When an item has been selected and recorded in the table, the text of the selected item is changed from black to blue. This gives the interviewer a visual marker that the item has already been selected. As previously described in screen design, the question text is 'grayed' when it is optional to read aloud to the respondent.

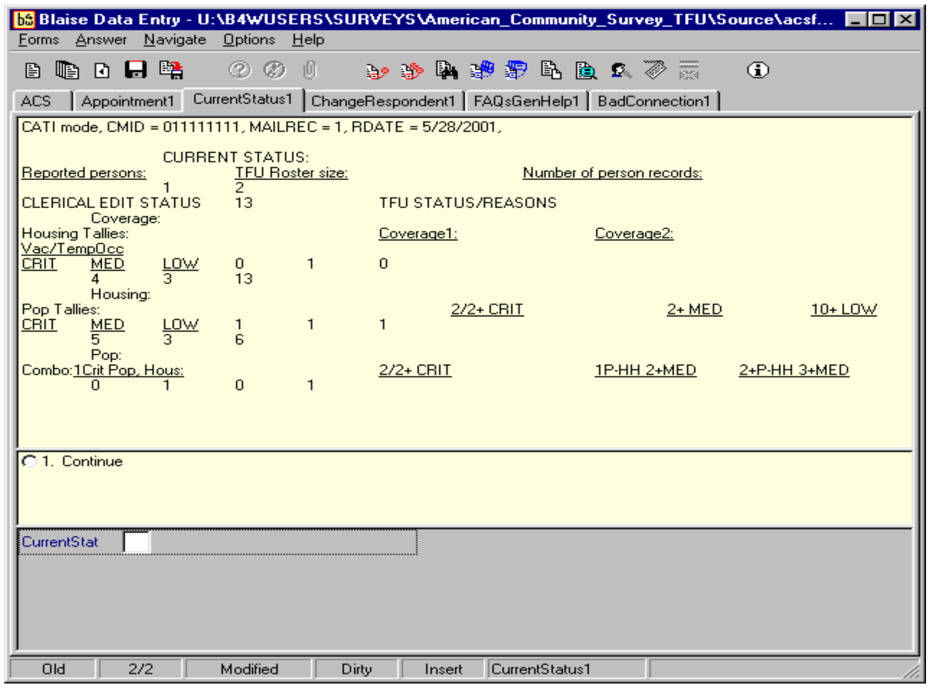
Open Grids

An open grid provides interviewers the flexibility to enter data in either a person based columnar style or in a topic based row style using a table. This method was developed in conjunction with Westat for the American Community Survey's Telephone Follow-Up instrument. This method as illustrated in Figure 9, was developed out of the need to provide the interviewer with the option to enter data either across a row or down a column and yet not slow down the data entry process. The data entry table structure enables the application of specific edits at the field level and the postponement of other error check edits until after the table is indicated as completed by the interviewer.

At the field level, edits are performed for selected fields such as age, e.g. where for example a value of 150 will trigger an immediate edit. Edits that are postponed until the interviewer indicates that they are finished collecting data for the table would be checks for empty fields. Using the EMPTY attribute on fields within the table allows the interviewer the flexibility of bypassing on route fields in a row and skipping around to other fields in the grid. At the point when the interviewer considers the table grid is complete, a "yes" is entered for finished and control is transferred to a block that performs edits on selected fields within the grid. If an edit is fired, the interviewer is returned to the field involved in the edit on the table. The edits are performed and data is entered or corrected on a field by field basis. Once the data is 'cleaned up,' control is transferred out of the table.

Current Status Block

In the American Community Survey Telephone Follow-Up project, a method was needed to determine the status of the entire data form upon demand. While flags could be set in the block to determine the status, the problem was that the results of the edit flags could not be tallied directly from the blocks where the flags were set. This is because the Blaise rules determine whether a field is 'on route' or 'on path' at any point in time. It is possible that a field that is currently 'off route' would not be included in the tally if the tally is computed at the field level. The current status block was developed in conjunction with Westat to address accurately calculating the field pass/fail flags at any point throughout the interview. This block calculates only the values of the flags and posts the error tallies by level of criticality and block identifier. The field based status flags in the instrument are constantly updated in the Current Status block. Since they are isolated, the RULES for all of the fields in all of the blocks are not run in order to calculate the totals. The range of the flag's error status as shown in Figure 9 includes: low, medium, critical or no errors (implied by zero totals in the categories). This internal design of a dedicated block also improved performance in the instrument.


Blaise Data Entry - U:\B4WUSERS\SURVEYS\American_Community_Survey_TFU\Source\acsf...
Forms Answer Navigate Options Help
ACS
Appointment1
CurrentStatus1
ChangeRespondent1
FAQsGenHelp1
BadConnection1

```
CATI mode, CMID = 011111111, MAILREC = 1, RDATE = 5/28/2001,

                   CURRENT STATUS:
Reported persons:         TFU Roster size:                Number of person records:
                   1      2
CLERICAL EDIT STATUS      13                TFU STATUS/REASONS
         Coverage:
Housing Tallies:                            Coverage1:                Coverage2:
Vac/TempOcc
CRIT     MED       LOW    0        1        0
         4         3      13
         Housing:
Pop Tallies:                                   2/2+ CRIT                2+ MED         10+ LOW
CRIT     MED       LOW    1        1        1
         5         3      6
         Pop:
Combo:1Crit Pop, Hous:                      2/2+ CRIT                1P-HH 2+MED    2+P-HH 3+MED
         0         1      0        1

◯ 1.  Continue

CurrentStat

Old   2/2   Modified   Dirty   Insert   CurrentStatus1
```

Figure 9

Performance Enhancement Strategies

Considering the configuration of the development platform, deploying Blaise instruments can require a large amount of cache memory. One of the most important requirements of interviewers is that they are able to collect data quickly especially when interviewing impatient and reluctant respondents. This requirement considers not just the length of the questionnaire but the ease of use of the instrument and its processing performance. Some survey instruments require many features and enhancements because of complex data collection issues. Extensive requirements can contribute to overhead processing which reduces the performance of the instrument. These requirements and the large memory demands of GUI based instruments drive the demand for high performing computers. A constant struggle in any technical organization is maintaining the most advanced technology available. This is especially true in the government sector where the acquisition of large numbers of computers can take many months and even years. The reality of most organizations is that the technology actually used will not always be the very latest. The Census Bureau fields over 6,000 laptops. Constantly maintaining the very latest technology is simply not cost-efficient. As a result, it is exigent that the programmer authors design instruments in the most efficient manner possible to maximize the use of the available technology. This means going beyond the basic programmer efficiency strategies such as avoiding long if-statements, or placing the most common condition first in long if-else statements, etc.

In going beyond these basic strategies, Blaise has some features to assist in enhancing performance such as running Cameleon to determine generated parameters and thereby minimize their use in coding. There is also the Watch Window blocking checking feature. This feature enables the authors to verify if only the appropriate blocks are being checked on selected fields. The Consumer Expenditure Survey project is currently the largest project in the Census Bureau that has been developed in Blaise. The completed instrument contains over 360,000 fields. Addressing performance in an instrument of this size of has been especially challenging and has demanded even more performance enhancement strategies. Constant testing was done during development to identify effective strategies for improving the instrument's performance. These strategies include:

- Controlling the running of the Blaise rules. This has been one of the key strategies to improving performance in this instrument. This was accomplished by:
  - Making separate blocks for the fields and rules in multi-shared blocks. This separation enables more flexibility for when rules are run for specific sets of fields.
  - Applying 'gates' to large blocks of code. The gate requires the interviewer to put a block or section 'on route' only when needed as shown the code below:

```
Fields
   Intro  "Now I am going to ask about the purchase or rental of household items."
          : (Continue  (1) "Enter 1 to continue",  NODK,  NORF)
Rules
  IF (Intro = Continue ) THEN
    {**** [ laundry list sections ] ****}
           Sect06
           Sect07
           Sect08
           Sect09
   ENDIF
```

- Structuring survey sections and arrays that minimize pagination to improve performance in the initial setup process.
- Consider performance impact when using external files such as edit range lookups.
- Use LOCALS only for looping.
- Set any external files to read-only whenever possible.
- Avoid KEEP statements for entire blocks when possible.

Recently in performance testing and benchmarking, a significant performance refrain was found in the CE instrument. For the current laptop configuration that will be deployed in production, the performance threshold was significantly impaired when the instrument exceeded 240,000 fields. This prompted the authors to do some internal redesigning and explore alternatives. Currently being considered is the implementation of two separate instruments. This was possible because of the unique character of the CE survey. One instrument could contain a read-only database that holds dependent data from previous interviews for the main instrument.

Lessons Learned

While there have been many lessons learned from the surveys that were authored by the TMO authoring staff only several key items can be delineated in this limited document:

- High level requirements should be carefully reviewed as they relate to the structures of the language that will be used to implement the requirements in the early stages of a project. This enables appropriate designing of an instrument to support required functionality considering the constraints of the software as well as efficiency and performance in the instrument. At this stage repeating structures can be identified which can lead to reusable or standard applications within the context of the survey.

- The identification of structures to refine with prototypes greatly reduces the risk of creating inefficient data structures. Prototyping design methodology identifies potential problems in the proposed design at the early stages of development.
- Established standards should be reviewed throughout the process so that their use and implementation is ensured. Standards that conflict with functionality or known limitations should be renegotiated without sacrificing the goals of the users.
- A high level flowchart is essential to good development design. It helps document and identify intended functionality. Often reusable or standard structures (blocks) can be identified.

In summary, obtaining extensive documentation early in the planning stages of the project and assisting the sponsor in the survey design is most conducive to good programming practices, design, and implementation that ultimately produces the best and most accurate data collection instrument.

Suggestions for Future Blaise Releases

Some of the design strategies discussed in this paper could be included as features in future Blaise releases. Listed below are some suggestions for development of future Blaise software releases.

- Section labels and column headings including options for font type, size and colors should be included as an option when defining grids in the modelib. Also, more rich text format options for the description area of the grid as well as in other places. This would eliminate the need for section bitmaps.
- A feature that evaluates the status of on-path and off-path data. This is especially useful when trying to control the run of the Blaise rules or to tally the status of flags as described in the current status block strategy.
- Expand the edittype feature to enable the field to be passed as a parameter.
- Make the autohide feature a default setting. Currently empty items are displayed as illustrated in Figure 7.
- Enhance the Blaise editor with features for: file comparisons, customized formatting options on selected code such as bold, italics, color, etc.
- A command to perform KEEPS on *all* elements of an array simultaneously.
- A dialog box during compilation that notifies the developer of:
    - New or modified parallel block text descriptions in the .bxi.
    - New or modified types in a previously generated .bxi.
    - The masking format on newly added types in the .bxi.

As more surveys are developed in the Blaise programming language the demand for these features and other will grow but the current functionality of Blaise is exceedingly flexible and enables considerable creativity in developing instruments that collect quality data.

Acknowledgements

Special thanks to Mark Pierzchala (Westat) for his consultation expertise and prototype assistance; Roberto Picha (Census Bureau) for his testing and performance enhancements; William Dyer, Jr. (Census Bureau) for his benchmarking, and all of the TMO Blaise Authors (Census Bureau) for their fine development work that led to many of the examples and conclusions presented in this paper.