Alternate order of blocks for the CPS-ASEC Instrument

Roberto Picha, U.S. Census Bureau

1. Introduction

The Current Population Survey's Annual Social and Economic Supplement (CPS ASEC) generates widely used estimates on detailed income data and benefit programs. Concerns of "fatigue effect" were raised in regards to the way the information was collected. This fatigue may lead respondents to lose concentration or motivation when answering long questionnaires, affecting the reliability on the data collected. After researching and completing cognitive testing on paper, it was determined that an alternate approach should be tested in the automated instrument. The idea was to collect age and incomes related information first and then tailor the order of detailed income questions based on the information collected. Instead of asking the questions in the benefit blocks in the same order for all respondents, the sponsor wanted to alternate the order in which the blocks were asked to optimize the responses. Three alternate paths were identified: low income, seniors, and all others. This paper discusses the implementation of a research paper and its findings into an automated instrument.

The Current Population Survey (CPS), sponsored jointly by the U.S. Census Bureau and the U.S. Bureau of Labor Statistics (BLS), is the primary source of labor force statistics for the population of the United States. The CPS is the source of numerous high-profile economic statistics, including the national unemployment rate, and provides data on a wide range of issues relating to employment and earnings. The CPS also collects extensive demographic data that complement and enhance our understanding of labor market conditions in the nation overall, among many different population groups, in the states and in substate areas.

2. Background

In March 2013, the Technology Management Office (TMO) authoring team began working with the Income Source Sections and converting them in an alternated order for CPS's ASEC. The new requirement was the result of the research conducted by Westat and documented in their paper "Cognitive Testing of Potential Changes to the Annual Social and Economic Supplement of The current Population Survey." The instrument went into production during February-April 2014; and will return to production in February-April 2015.

The CPS-ASEC instrument administers up to twenty-three possible sections for participants in a household composition for ages 15 and above, and concerns of "fatigue effect" were raised in regards to the way the information was collected in the instrument. Westat's paper resulted in cognitive testing which in turn prompted the CPS sponsor to implement this alternate approach in the current automated instrument. The idea was to collect age and household level income related information first and then tailor the order of questions based on the information collected. Instead of asking the questions in the benefit section blocks in the same order for all respondents, the instrument would administer them in the order the research indicated was most appropriate for the respondent (alternate order). The order was based on information already collected and would optimize the responses.

The Table 1 below displays the order of the instrument sections based on the income type. The table shows the default, low income, and senior paths along with the order of the sections within each path.

Table 1. Alternate Order of Income Source Questions

Default		Low Income		Seniors	
Order		Order		Order	
1	Unemployment and Workers Compensation	1	Unemployment and Workers Compensation	2	Disability
2	Disability	7	Public Assistance / TANF	3	Social Security
3	Social Security/SS for Children	8	Food Stamps (SNAP)	4	Supplemental Security Income (SSI)
4	Supplemental Security Income (SSI)	9	WIC	5	Veterans
5	Veterans	10	School Lunches	6	Survivors
6	Survivor Benefits	11	Public Housing	13	Pensions
7	Public Assistance / TANF	12	Energy Assistance	14	Annuities
8	Food Stamps (SNAP)	2	Disability	15	Retirement Accounts – Withdrawals or distributions
9	WIC- no amount collection	3	Social Security	16	Other Income Earning Assets (outside of retirement)
10	School Lunches- no amount collection	4	Supplemental Security Income (SSI)	17	Property Income
11	Public Housing- no amount collection	5	Veterans	1	Unemployment and Workers Compensation
12	Energy Assistance	6	Survivor Benefits	7	Public Assistance / TANF
13	Pensions	13	Pensions	8	Food Stamps (SNAP)
14	Annuities	14	Annuities	9	WIC
15	Retirement Accounts (within) – Withdrawals or distributions	15	Retirement Accounts – Withdrawals or distributions	10	School Lunches
16	Other Income Earning Assets (outside of retirement)	16	Other Income Earning Assets (outside of retirement)	11	Public Housing
17	Property Income	17	Property Income	12	Energy Assistance
18	Education Assistance	18	Education Assistance	18	Education Assistance
19	Child Support	19	Child Support	19	Child Support
20	Alimony	20	Alimony	20	Alimony
21	Financial Assistance from friends or relatives	21	Financial Assistance from friends or relatives	21	Financial Assistance from friends or relatives
22	Other Income I (as in ASEC now)	22	Other Income I (as in ASEC now)	22	Other Income I (as in ASEC now)
23	Other Income II (as in ASEC now)	23	Other Income II (as in ASEC now)	23	Other Income II (as in ASEC now)

3. The goals for the CPS-ASEC redesign

The initial stage of the CPS-ASEC redesign, from the standpoint of specifications, suggested following the Research Triangle Institute concept that was implemented in the Survey of Health Insurance Program Participation (SHIPP) instrument using a global counter flag to trigger specific paths. In other words, the specifications for the CPS-ASEC redesign would be defined for each section for each of the three paths using the Specialized Instrument Design Resource (SPIDER) application. This tool is used to quickly create an instrument specification in the U.S. Census Bureau.

Furthermore, from the beginning the intent was to reuse the existing code of the ASEC as much as possible. The sponsors hoped that TMO authoring would make small internal modifications in the code. While this was a great goal, the authoring team later learned that some blocks would have to be split and parameters would need to be modified. This was initially a concern that later changed once there was a better understanding of the task.

With these two goals in mind, the authors developed a small prototype to assure the sponsoring division that one set of specifications could be used to simplify the process of managing the questions despite the path taken. And, more importantly, to determine if the alternate order of sections would be feasible.
There were additional changes in the sections due to new questions being added as well as the removal of other questions. Parameters no longer in use had to be removed. Data dependencies across sections were considered and modifications were expected.

3.1 A Simple approach sometimes is not that simple

The TMO authoring team looked into the routing groups containing the sections. However, there was a need to come up with a proof of concept for placing sections in specific order based on some universe. The twenty-three sections were classified into five groups (see Table 2 for the grouping suggested).

For the first attempt, the authors created a small-scale instrument; it mimicked the groups on route. Table 2 summarizes the groups containing the sections

Table 2. Grouping sections
○ Group A Section 1 ○ Group B Sections 2,3,4,5,6 ○ Group C Sections 7,8,9,10,11,12 ○ Group D Sections 13,14,15,16, 17 ○ Group E Sections 18,19,20,21,22,23

Eventually, Section 1 was removed from Group A. The questions for this section were removed or placed into other sections. Therefore, there were only four groups that had to be re-ordered to meet the new requirements for the CPS-ASEC instrument.

Table 3 shows the three paths for the four groups. This grouping helped to simplify the final structure of the instrument, and at the same time it helped to identify the issues up front.
Group E was always asked at the end regardless of the path taken. In essence, the alternate order would be between Group B, C, and D.

Table 3. Identifying the path

- Default path: B, C, D, E
- Low Income path: C, B, D, E
- Senior Path: B, D, C, E

Table 4 shows a simple Datamodel that was used to prototype the concept by routing four variables representing the four groups to simulate how the sections would be used in the final approach. This simple exercise demonstrated to our sponsor that the reuse of existing blocks was not as simple as expected. The re-ordering was not a mere one step modification in the instrument that could simply be tested and placed into production.

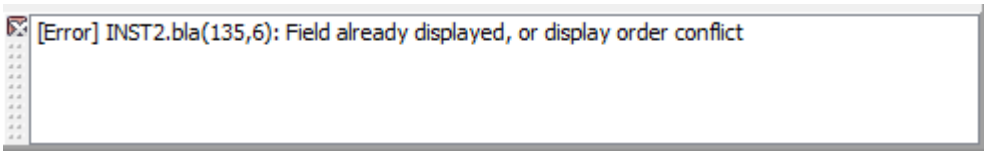
Table 4. Simple Datamodel

```
DATAMODEL Inst "Routing questions"
FIELDS
B, C, D, E: STRING
PATH : ( D "default", L "low Income", S "Seniors")

RULES

PATH
IF PATH = D THEN
   B
   C
   D
ELSEIF PATH = L THEN
   C
   B
   D
ELSE
   B
   D
   C
ENDIF

   E
```



As any savvy Blaise author would guess, the first error thrown by the compiler was "*Field already displayed, or display order conflict*". This message means that you cannot route the sections in different order.

Routing questions in Blaise can be challenging, especially if the order might vary at runtime. The restriction is that the layout cannot be changed at runtime.

Other techniques could be applied to obtain the desired result, but these techniques would not be automated. For example, parallel blocks could be used in the instrument, and the interviewers could manually select the appropriate section to interview based on the order required. The sponsors did not want interviewers determining the order in which to interview blocks. So, they proceeded to develop specifications in SPIDER to create three separate blocks, each representing one of the question order treatments.

3.2 Prototyping the alternate blocks in three paths

A second small Datamodel prototype was developed using a technique conducive to achieving the alternate order of income source questions. Expanding the Datamodel to contain the 23 sections and wrapping the sections into five defined groups (as previously described) made more sense. This also helped the analysts to visualize the future structure of the instrument.

The prototype established the concept of using a master block. The master block was the location where data was stored from any of the three paths taken in the case. This master block would aid the analyst during testing to verify that data was stored in one place and always the same place regardless of the path taken. It would also simplify the back-end data processing.

Table 5 displays the block structure for the supplement. While not fully functional for data collection, the prototype served as a model for our sponsors so they could develop the specifications required for this instrument. Essentially, the introduction of the master block concept allowed the sponsor to developed one set of specifications, and the approach was cleaner. The dependency across sections was a concern. However, that eventually subsided as the instrument for the most part was referencing variables directly as opposed to being properly parameterized.

Table 5 - Datamodel displaying the blocks

```
DATAMODEL Inst
 FIELDS

   Path : (Default "B,C,D,E", LowIncome "C,B,D,E", Seniors "B,D,C,E")

 BLOCK BlkB  ;   BLOCK BlkC  ;   BLOCK BlkD  ;   BLOCK BlkE

 BLOCK BOrderOne   { default order }
   FIELDS       B : BlkB;      C : BlkC;     D : BlkD;       E : BlkE
 ENDBLOCK

 BLOCK BOrderTwo  { low Income order }
   FIELDS       C : BlkC;     B : BlkB;     D : BlkD;       E : BlkE
 ENDBLOCK

 BLOCK BOrderThree  {Senior order}
   FIELDS       B : BlkB;      D : BlkD;      C : BlkC;       E : BlkE
 ENDBLOCK

 BLOCK BMaster {master copy of the answers (no matter which order)}
   Fields       B : BlkB;      C : BlkC      D : BlkD;       E : BlkE
 ENDBLOCK

 AUXFIELDS
   OrderOne    : BOrderOne
   OrderTwo    : BOrderTwo
   OrderThree  : BOrderThree
   DoOnce : 0..1

 FIELDS
        Master : BMaster
```

This new structure more accurately resembled the final product. This new concept would give us the following benefits:

- **Centralized location of data** - The data collected from these various blocks would reside in a "master block". Here the data processors could extract the data items from one location.
- **Minimize Database Size -** Using temporary blocks for data collection will not add any more space in the survey database.
- **Allows for Rostering -** The prototype was designed for one person. However, it was later adjusted to handle a household roster by adding a roster to loop through the household members.

Table 6 shows the routing and synchronization of the blocks. The first step was to copy data from the master blocks into the appropriate routing blocks. After data was collected in the routing blocks, it was copied back into the master blocks.

Table 6. Displaying routing and assignments

```
RULES
  Master.KEEP    {keep the master set of answers}
  path
  DoOnce.KEEP
  IF DoOnce = EMPTY THEN     {use block copy to transfer master responses to the block to be used}
     IF path.ord =1 THEN
         OrderOne.B := Master.B
         OrderOne.C := Master.C
         OrderOne.D := Master.D
         OrderOne.E := Master.E
    ELSEIF path.ord =2 THEN
         OrderTwo.C := Master.C
         OrderTwo.B := Master.B
         OrderTwo.D := Master.D
         OrderTwo.E := Master.E
     ELSEIF path.ord = 3 THEN
         OrderThree.B := Master.B
         OrderThree.D := Master.D
         OrderThree.C := Master.C
         OrderThree.E := Master.E
     ENDIF
     DoOnce :=1
  ENDIF

  IF path.ord=1 THEN                {ask the blocks in alternate order}
       OrderOne
       Master.B := OrderOne.B A     {use block copy to transfer the responses back to master}
       Master.C := OrderOne.C
       Master.D := OrderOne.D
       Master.E := OrderOne.E
  ELSEIF path.ord=2 THEN
       OrderTwo
       Master.C := OrderTwo.C        {use block copy to transfer the responses back to master}
       Master.B := OrderTwo.B
       Master.D := OrderTwo.D
       Master.E := OrderTwo.E
  ELSEIF path.ord=3 THEN
     OrderThree
     Master.B := OrderThree.B     {use block copy to transfer the responses back to master}
     Master.D := OrderThree.D
     Master.C := OrderThree.C
     Master.E := OrderThree.E
  ENDIF
```

3.3 Final steps before implementation

Only one set of specifications in SPIDER was necessary for any path taken by the instrument.
The sponsor agreed to one important business rule. Once the path was established during data collection, the interviewer was not allowed to back up and modify data that would change routing. This particular rule was stressed during training. Although, there is a solution for situations where an interviewer may change the path, it was preferred not to implement this solution as the enhancement might have added a performance issue to the instrument.

4. Implementation

Once the sponsor was satisfied with the prototype, the next step was to implement the prototype methodology into the CPS-ASEC instrument by modifying the existing code. Each of three groups containing the twenty-three sections was self-contained, but a few of them had relationships across sections. This imposed some challenges that required additional code modifications. Some of the modifications needed were:

- Changes to add and remove parameters.
- Splitting blocks
- Referencing arrayed variables directly
- The removal of fill constructs
- Layout in the modelibrary was adjusted as new questions were added

4.1 Re-use of existing code

Since the CPS-ASEC was already coded, the authors quickly understood that they needed to break some larger blocks into smaller blocks to achieve the instrument's proper routing of questions.

Examples of the changes made in the call of sub blocks are shown below.

- **Old Code:**
  - **Call to source**
    bVet_Pmt({IMPORT} CTRLNUM, IN_NUMHOU, IN_RESPLI, IN_NAME, IN_SEX, IN_LAST_YEAR)
  - **Person Level:**
    bVet_Pmt_Person[I]({IMPORT} IN_CTRLNUM, I, IN_RESPLI, IN_NAME[I], IN_SEX[I], IN_LAST_YEAR)

- **New code:**
  - **Call to source**
    bVet_Pmt({IMPORT} CTRLNUM, IN_NUMHOU, IN_RESPLI, IN_NAME, IN_SEX, IN_LAST_YEAR)
  - **Person level:**
    bVet_Pmt_Person[I] ({IMPORT} IN_CTRLNUM, I, IN_RESPLI, IN_NAME[I], IN_SEX[I], IN_LAST_YEAR)
  - **Call to amount:**
    bVet_Pmt_Amt({IMPORT} CTRLNUM, IN_NUMHOU, IN_RESPLI, IN_NAME, IN_SEX, IN_LAST_YEAR, IN_Q60a88, IN_Q60b_88)

As you can see, there were really no differences between the old method and the call to the source blocks in the new method.

The only changes occurred in the call to the Amounts blocks when using the new path. Additionally, inside the person level blocks in the new method there were many instances where new parameters were generated (GPs). This is because the fields that needed to be accessed were person level source fields and passing them would mean passing (in the case of veterans questions) four parameters per person and in some blocks more. In the old method, this was not necessary since the fields were at the same level.

4.2 Unexpected Issue

As the work on the redesign took shape, one unexpected major issue made us change the approach. Data was not being correctly saved to the Auxfield blocks as expected.

During our system test, the subject matter team observed that when exiting the case in the middle of the new income sections, data was lost. While data was stored properly in the master block, it was noted that upon re-entry of a previously touched case, the data did not appear to copy into the Auxfields blocks. This occurred particularly in the Amount blocks where the logic relied on data from the source block to be on route. However, because of the way Blaise works (Select Check Mechanism), it was not properly evaluating the parameters referenced causing the amount data to be thrown off path.

Because there were several sections and blocks involved, the amount of work it would take to correct the issues by redesigning the communication between sections was significant. An executive decision was made to make a more conservative change in the instrument. Instead of using Auxfields for the alternate order of blocks, these Fields would be converted into regular Fields.

This change brought some consequences for the post collection processing stage.

4.3 Comparison of Meta file

Since the Auxfields blocks were converted to Fields blocks, the size of the instrument also increased. Now the CPS ASEC instrument became the largest Blaise instrument compiled in the U.S. Census Bureau. To have an idea in the increment, the size of the metafile went from 2,856KB to 18,764 KB.

4.3.1 Technical description of the model instrument

The following tables (7 and 8) show the differences of the metafile size before and after the redesign. The increase in the size is significant. Using Cameleon and the technical description of the metafile for the CPS-ASEC before and after the redesign shows the increment of the metafile size.

Table 7: **Overall counts**	**Values**		**Percent**
	ASEC 2013	ASEC 2014	Increase
Number of uniquely defined fields*1	2,473	5,681	229.7%
Number of elementary fields*2	2,329	5,182	222.5%
Number of defined data fields*3	24,623	69,131	280.8%
Number of defined block fields*4	144	499	346.5%
Number of defined blocks	233	368	157.9%
Number of embedded blocks	9	9	100.0%
Number of block instances	1,570	4,580	291.7%
Number of key fields	1	1	100.0%
Number of defined answer categories	928	1,321	142.3%
Total length of string fields	191,205	396,662	207.5%
Total length of open fields	0	0	na
Total length of field texts	143,863	424,282	294.9%
Total length of value texts	43,343	50,153	115.7%
Number of stored signals and checks	10,069	25,428	252.5%
Total number of signals and checks	10,069	25,428	252.5%

Table 8: **Data fields**	ASEC 2013		ASEC 2014	
	Number	Length	Number	Length
fields in data file				
Integer	11167	39582	28466	116226
Real	323	2213	323	2213
Enumerated	7346	8811	23216	25769
Set	507	790	3523	6198
Classification	0	0	0	0
Datetype	53	424	53	424
Timetype	3	24	3	24
String	5224	191205	13547	396662
Open	0	-	0	-
Total in data model	24623	243049		

5. Struggles in Post-Collection Processing CPS-ASEC

The Current Population Survey Programming Branch (CPSPB) was an active participant in the process of the new redesign. For the purpose of data output, the sponsoring division defined the specifications for these sections only once and not three times. In the initial design of the instrument, with the use of temporary Auxfields, data processing appeared to be straightforward. However, due to the later change that required Auxfields to be converted to Fields, data processing became much more complex. The CPSPB soon discovered that there were duplicates of all their records. The duplicate records appeared in the raw data. Upon reviewing the data, it was noted that the duplicates were instances of the blocks and the instance was not always the same one. Because this was unexpected, CPSPB was puzzled and unsure when extracting the data from case to case. The reason behind this duplicate data traced back to the blocks that were reused for data collection and storage. Therefore, their instance would be reflected in output. The first, second, and third instance would represent the data of the path used for the case. The fourth (last) instance was in fact the master block and its data.

The CPSPB followed specifications from SPIDER that was used to process the CPS-ASEC instrument. Below is a description of the method used. The main task was identifying the blocks that had these duplicates in them. These references, (i.e., instances of the blocks) were found during an internal 50 case test. This test is a step used internally to validate the data before the instrument is deployed to production. The method applied to validate the test is also the same method used for production data processing.

The following is a high level description of what the CPSPB were looking at. These are the files from an ASCII Relational dump.

- CPS-ASEC test output consists of many text files (labeled inst.A01, inst.A02, etc), called blocks.
- Each block is either a person level block (consists of person records) or a household level block (consists of household level data).
- Each block has a different file layout, which is documented in our .RAS file. However, the first 13 columns for EVERY block are identical:
    - FPRIMARY 1 – 8
    - INSTANCENUMBER 9 - 13

- Fprimary is the record ID and Instance Number represents the income path taken.

- For Household Level blocks:
    - 1 = default income path
    - 2 = low-income path
    - 3 = senior income path
    - 4 = output record (can represent any path)

- For Person Level Blocks:
    - 1-16 = default income path
    - 17-32 = low-income path
    - 33-48 = senior income path
    - 49+ = output record (can represent any path)

- There was a misunderstanding of records with an Instance Number = 1, 2, or 3. The same information from one of these records was duplicated in Instance Number 4 (household level blocks). The data processors were not sure where to retrieve the data from. In this case, the record is duplicated (two identical records in a block exist, and only their Instance Number is different) so the data processors were instructed to always use the Instance Number 4 for retrieving the household level block data.

- The same output was noted for Person blocks, except that now the Instance Numbers will be 1-16, 17-32, 33-48, and 49+. The data processors were instructed to retrieve their data from Instance Numbers 49-64 for the person level output records.

5.1 What CPSPB would want instead?

The CPSPB that processes the CPS data desired a slight change in the Blaise output to produce simpler, more straightforward data in their ASCII Relational dump - specifically the data of the instance number of the path taken for the case.

For household blocks, the data output team had to remove all records that had an Instance Number = 1, 2, or 3. These are all duplicates of the Instance Number=4 records.

For person blocks, the data output team had to remove all records that had Instance Number < 49. These are all duplicates of the Instance Number 49-64.

For example:

Household level block B03

Fprimary	InstanceNumber	
00000001	**1**	**← remove**
00000001	4	
00000002	**2**	**← remove**
00000002	4	
00000003	**2**	**← remove**
00000003	4	

Person level block B34

Fprimary	InstanceNumber	
00000001	**1**	**← remove**
00000001	49	
00000002	**17**	**← remove**
00000002	50	
00000003	**29**	**← remove**
00000003	49	

To meet the requests of the data output team, we could reinstate block Auxfields with some other modifications. Then only one Instance would be present in the data for the household level and one for each person level, regardless of the order of collection.

5.2 How Data is handled in Output

Original raw data are not actually removed from records; the CPSPB removes the records as they process it. The sponsoring division tells the CPSPB which blocks these duplications are occurring in and provides instructions on which records to remove via specifications.

This step tells the CPSPB which blocks have duplicate records. However, for household blocks, the CPSPB has to go a step further. They must use the .RAS file to find out where the inst.xxx blocks are in SPIDER. For example, see below.

B33: blkUWComp_Source
B35: blkinc_Source_A
B50: blkInc_Source_B
B54: blkInc_Source_C
B64: blkInc_Source_D
B71: blkUWComp_Amt
B73: blkInc_Amt_A
B74: blkDis_Incom_Amt

Since the sponsor uses SPIDER to produce a list of variables it wants on output via physical location, this location matches the RAS file produced. As explained earlier, the extra instances threw off the post data collection processing.

5.2 Master Control System (MCS) struggles in the process of recycling cases

Another issue was found during production. Our Master Control System (MCS) experienced issues when running a Manipula recycle script against the CPS ASEC metafile. The script was used to recycle cases from CATI to CAPI and vice versa. When the MCS ran the script, an access violation crashed their process. The instrument and scripts were compiled using Blaise 4.8.1 build 1403. Per Statistics Netherlands advice, MCS modified their process to use Blaise 4.8.4 build 1861 Manipula to process the Blaise 1403 Manipula script. This change resolved the access violation. While we were not able to identify the actual issue that caused the crash, TMO and the sponsoring division decided to move the instrument to a newer version of Blaise for 2015 production.

6. Next Steps

The CPS-ASEC is currently in another round of interviewing for 2015. There were no major changes made to the ASEC other than the typical changes such as text, fills, and adding/removing questions. The structure established from the Redesign implemented in 2014 will be similar for 2105. However, the CPSPB has modified their process to read the proper instance blocks in order to simplify their processing.

- The 4^{th} instance for house hold data
- The 49^{th} + instance for person level data

6.1 Going back to Auxifields

Late in 2014, TMO Authoring attempted to revert to Auxfields blocks with few adjustments to the original approach. The initial findings indicated that some parts of the income blocks may lose data. This happened to all blocks that were arrayed. Adding some Keep statements might have provided a solution. However, there was not sufficient time to test this change adequately. We plan to research a solution for this issue for the 2016 data collection effort.

6.2 Manipula can be handy in the future

For 2016, we plan to synchronize the store and load using Manipula. The proposal is to insert the script in the instrument. We still need to add some keeps to the blocks that initially were not retaining the data. The goal is to reduce the size of the metafile and make the instrument more robust while still maintaining the quality of the data collected.

7. Conclusions

Although we were unable to implement the desired design for the CPS ASEC supplement requirements, we were still able to implement the desired requirements by our sponsors. Overall, the mission was accomplished, but not in the desired manner. We, of course, learned from this experience.

- It was much more challenging that expected to implement a new instrument design into existing instrument code. It was very tedious to break up existing income questions blocks into smaller blocks so they could flow as required.
- Larger instrument tend to generate new issues in production.
- Returning to the Auxfields approach by using Manipula and adding proper keeps to the code should help to reduce the size of the instrument and simplify the data processing by eliminating duplicate data.
- Output data should be reviewed and analyzed by the data processors in the development stage and not after production has started.
- Because the sponsoring division fielded the CPS ASEC instrument in 2014 by itself, there was no way to measure how the redesigned ASEC supplement affected the quality of the data. As a result, management decided to field a second instrument in March 2015 in parallel with the (redesigned) production data collection effort. The second instrument consisted of the 2015 basic CPS instrument with the old (i.e., pre-redesign) ASEC supplement module. The sponsoring division will compare the data from the two instruments and draw conclusions as to the effectiveness of the redesigned supplement.

9. Acknowledgment

I'd like to acknowledge the following authors and analyst for their input into this paper.

- Johanna Rupp, ITS specialist. TMO
- Lisa A Cheok, survey statistician. ADDP
- Tim Marshall, survey statistician. ADDP

10. References

Wendy Hicks and Jeffrey Kerwin, "Cognitive Testing of Potential Changes to the Annual Social and Economic Supplement of The current Population Survey," July 25, 2011.

Barbara S. Bibb, Lillie Barber and Ansu Koshy, "Coding Tricks to Save Resources, Research Triangle Institute (RTI)," USA IBUC 2010 13th International Blaise Users Conference

The views expressed in this paper are those of the authors and not necessarily those of the U.S. Census Bureau.