Blaise Documentation System

Peter Sparks and Youhong Liu, University of Michigan

1. Introduction

The University of Michigan (UM) has created a tool (BlaiseDoc) for producing XML-based codebooks and questionnaires, which may be printed or viewed as web pages. In the course of the development, the system was tested on several complex instruments. This paper describes the design of the system and the methods used in producing the automated documents. There are different levels of information stored in the XML document, including "go to" logic, fills, external lookup references, multiple languages, and help references. The XML can then be output using different stylesheets for different target documents. Document examples are provided.

This paper also describes problems encountered during development and future enhancements under consideration.

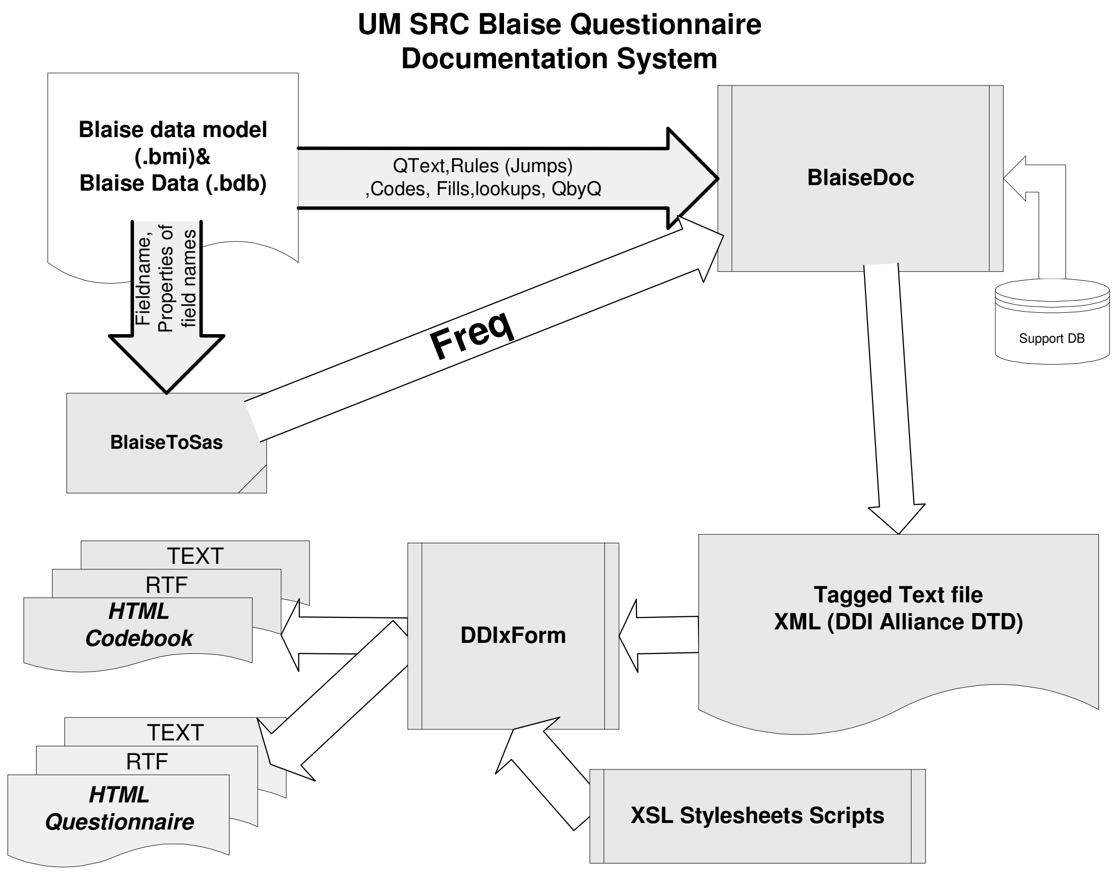
BlaiseDoc is a powerful tool to help create questionnaires, codebooks, and other documentation based upon the Blaise datamodel for a study. It works by analyzing the datamodel and its associated files, exporting that information to an XML file, and then displaying only what's needed via XSL (stylesheets) for the user.

2. Overall flowchart of system

BlaiseDoc has been designed to work as part of a larger system, parts of which are still being implemented. The current program reads a Blaise datamodel (and its associated external lookup files) and generates one XML file per language selected. It then transforms the XML data via a selected stylesheet to generate the output. This can be tailored to individual user needs by designing unique stylesheets that meet those needs.

The system has been designed to also work with SAS© data to read frequencies that will then be merged with the XML output. The frequencies had been done with Cameleon scripts, Blaise Manipula scripts, and a fair amount of hand processing. A new tool, BlaiseToSas, has been created to automate the export of Blaise data into a format more compatible SRC standards.

Other options include merging interviewer online help documents and respondent booklet information at the appropriate points in the generated document from a support database.


UM SRC Blaise Questionnaire Documentation System
Blaise data model (.bmi)& Blaise Data (.bdb)
QText,Rules (Jumps) ,Codes, Fills,lookups, QbyQ
BlaiseDoc
Support DB
Fieldname, Properties of field names
Freq
BlaiseToSas
Tagged Text file XML (DDI Alliance DTD)
DDIxForm
TEXT
RTF
HTML Codebook
TEXT
RTF
HTML Questionnaire
XSL Stylesheets Scripts

3. Methods used in producing automated documents

Creating a final document from a datamodel takes several steps and a great deal of processing. Visual Basic with the Blaise Component Pack was used for the main program. Additional modules were written (also in VB) so that a support database and SAS© could be included in the output. VB.net was used to implement the transformation of the DDI XML file into the different output files.

3.1 Processing the datamodel

BlaiseDoc begins by processing the entire datamodel per each language chosen. The output is dependent upon each language because there may be more or fewer questions, different fills, different frequencies and so forth for each. The program processes the datamodel by stepping through every rule through the DictionaryAsStatements method. It was found to be much faster than using the RulesNavigator for the same information and less memory intensive.

Each appropriate statement is added to a global structure in memory that annotates the datamodel. The different types of statements stored include

Questions:

ASK	if presented at some point they are stored as a normal question
assigned	if the question is not ASKed but is assigned a value they are shown as constructed.
SHOW	if not ASKed or assigned they are presented as a preload question
KEEP	if not ASKed or assigned they are presented as a preload question

Consistency checks:

CHECK	is presented as a hard consistency check
SIGNAL	is presented as a consistency check

Conditions:

IF	if a destination for a goto is present within the IF logic, and the IF logic itself cannot be expressed simply, then an internal checkpoint is created. "Simply" means an enumerated condition that can be resolved, such as A54 = Code1.

The execution of BlaiseDoc is essentially linear.

Action	**Description**
Collect user inputs to the program	Destination directory Datamodel to use External files Languages to process Stylesheet to use Support database (if used) SAS© data (if used) Blocks to process
Set up the directory structure	Needed for the output
Process external databases	Read in and write the data from the external databases to a separate XML file.
Step through datamodel in RULES order.	Add questions, consistency checks, internal checkpoints to a global structure in memory.
Examine each enumerated question and internal checkpoint	Resolve the destination goto for each value, DK, and RF.
Look for fill references in text (questions, codes, consistency checks)	Resolve each fill use
Create the XML file(s)	Write the information in the global structure and datamodel to an XML file per language selected. Place help and RB references contained in the support database into the question text. Merge frequency data from SAS© (if used).
Apply stylesheet	Current stylesheets produce HTML, RTF, and TXT output.

3.2 Notes on Resolving Goto references

The determination the destination of an enumerated or DK/RF value from a question is difficult. It requires a scan proceeding from the next statement after the question/internal checkpoint until an appropriate "goto destination" is found. The destinations can either be another checkpoint, another question that is asked, or the end of the Blaise datamodel.

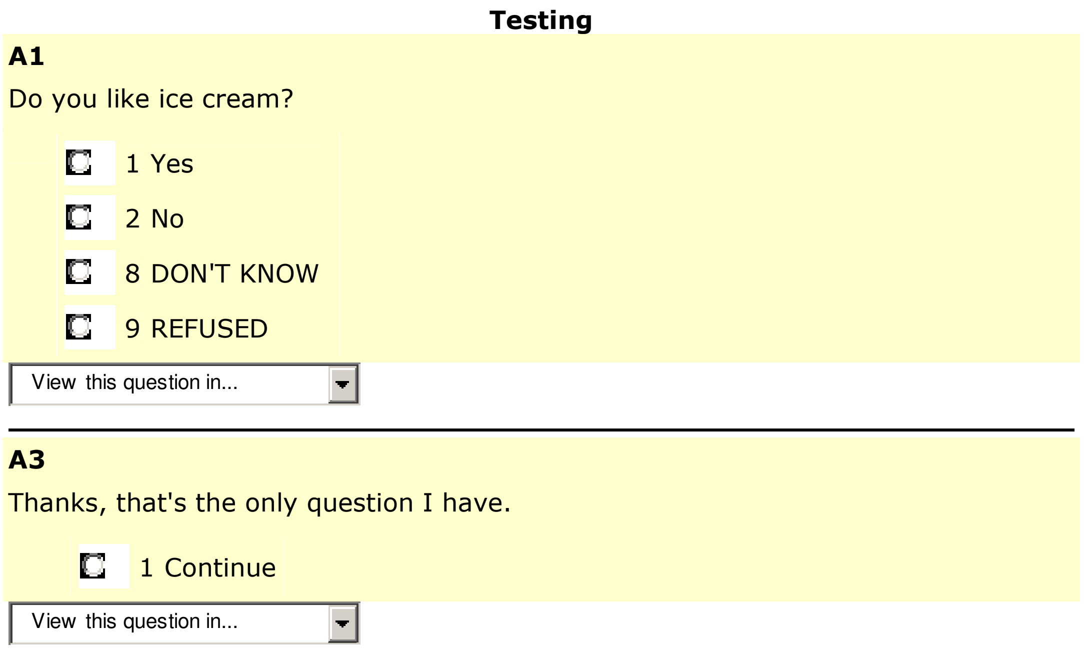
Each value/DK/RF for the question/internal checkpoint needs to be examined since it's possible to have different destinations for each. Each goto destination needs to be examined to see if it is an appropriate destination according to the logic and language. For example,

```
RULES
(1)     A1
(2)     IF ACTIVELANGUAGE = SPA THEN
(3)             IF A1 = Yes THEN
(4)                     A2S
(5)             ENDIF
(6)     ENDIF
(7)     A3
```

The questions on lines (1), (4), and (7) are added to the global structure as questions. The conditions on (2) and (3) are not added as internal checkpoints because both can be resolved simply. Line (4) is the first "goto destination" from A1, but it is not valid within the context of default language. Hence all goto's from A1 are to A3, line (7), for the default language.

If the language was SPA, then A2S is the first valid goto destination but only for A1 = Yes. All other values of A1 are to A3. The output from the above program is shown below.

3.2.1 English Sample Output



Testing

A1

Do you like ice cream?

- 1 Yes
- 2 No
- 8 DON'T KNOW
- 9 REFUSED

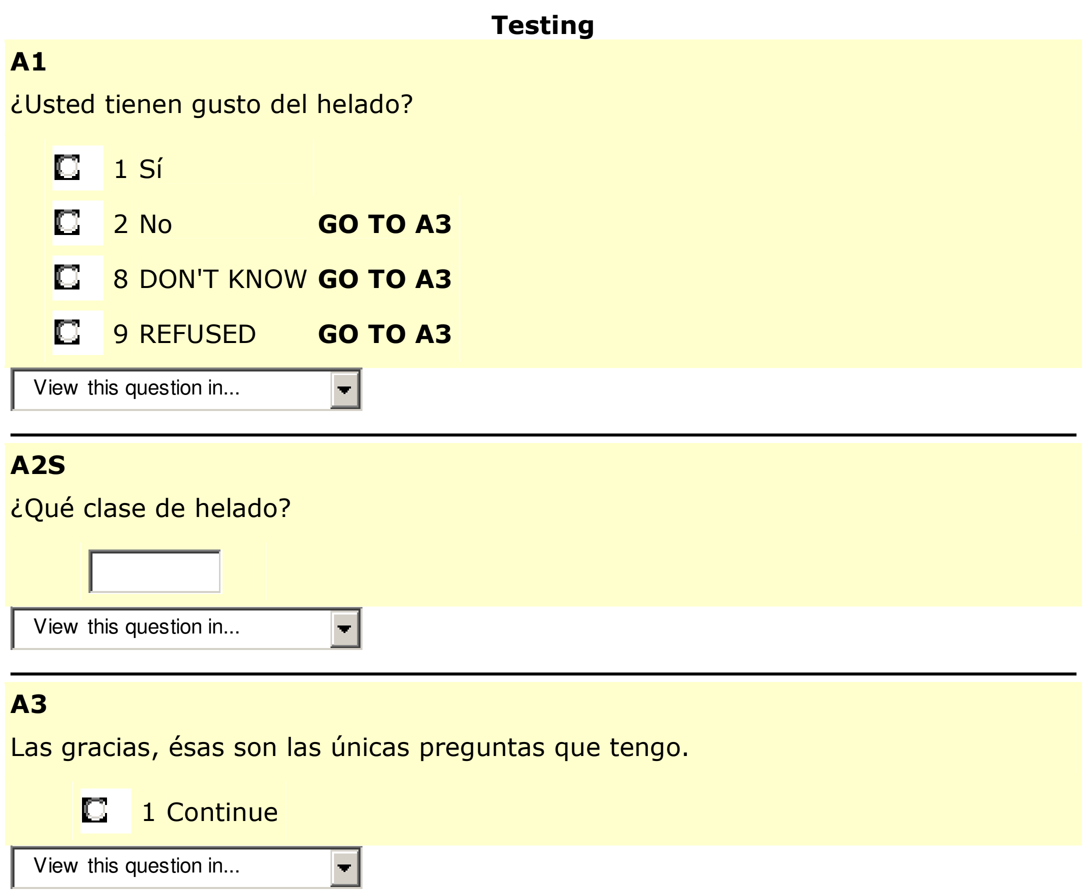
A3

Thanks, that's the only question I have.

- 1 Continue

3.2.2 Spanish Sample Output

The Spanish that is presented below was translated via AltaVista (http://babelfish.altavista.com/babelfish/tr) and is only intended for demonstration purposes.



Testing

A1

¿Usted tienen gusto del helado?

○	1 Sí	
○	2 No	**GO TO A3**
○	8 DON'T KNOW	**GO TO A3**
○	9 REFUSED	**GO TO A3**

View this question in...

A2S

¿Qué clase de helado?

View this question in...

A3

Las gracias, ésas son las únicas preguntas que tengo.

○ 1 Continue

View this question in...

3.3 Notes on Resolving fill references

Fills are resolved by a complex algorithm that works from the rules in the same block as the question. If assignments to the fill variable are not found in the context, Blaise doc backs up a level and searches the rules of each prior block until the fill is resolved. It will do forward searches to subblocks, and matches internal block parameters to parent parameters from the calling block. It will be unable to resolve a fill if the fill is assigned within a procedure or is a result of a external search. In this case the fill variable name is displayed.

3.4 Overview of XML content

XML (e**X**tended **M**arkup **L**anguage) was chosen as an intermediate output of the BlaiseDoc program. It gives a great deal of flexibility when exporting the information from processing into different formats based upon user preference. The initial XML output for BlaiseDoc was custom made for the program. This was changed to take advantage of a standard being supported by the DDI Alliance whose members include a number of organizations across North America and Europe (http://webapp.icpsr.umich.edu/cocoon/DDI-COMMITTEE/expert.xml).

3.5 DDI

The Data Documentation Initiative (DDI) is an effort to provide a standard for the documentation of datasets in the social and behavioral sciences internationally. The standard has an XML base that enables efficient use of the datasets.

The structure of the DDI is stored in a Document Type Definition (DTD) which provides a information about the elements, attributes, comment, notes, and entities that are used in the DDI.

More information about the DDI can be found at http://www.icpsr.umich.edu/DDI/index.html.

You can also download a copy of the DTD from
http://www.icpsr.umich.edu/DDI/users/dtd/index.html#version2.0.

Information on the DDI Alliance and the DTD were found on ICPSR's website. ICPSR is the Inter-University Consortium for Political and Social Research at the University of Michigan.

3.6 Stylesheets

The final step of the process involves running stylesheets against the exported XML data. This process is straightforward and makes use of VB.Net to provide the transformation of the XML data into RTF and HTML outputs. This transformation is performed by DDIxForm, a VB.net program that is called from BlaiseDoc (a VB6 program). There are plans to move the program to just one platform and so DDIxForm will be incorporated into the BlaiseDoc program.

4. Installing BlaiseDoc

BlaiseDoc requires a series of installations in order to work. They include

BlaiseDoc
Blaise BCP 2.0
DDIxForm
MS_XML_4.0
SAS© (if working with SAS© data)

It is highly recommended that the current program is installed on Windows XP (or later) machines only. As part of the packaging process certain system DLLs are incorporated as part of the setup. When the install routine is run on an older Windows operating system it may attempt to update key system files with those from XP. These are incompatible with the other system files on the older system.

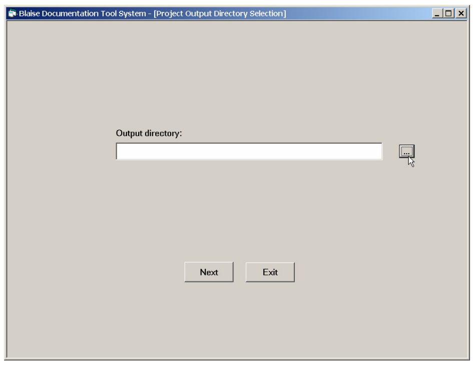
If working with SAS© an additional ODBC connection needs to be established to the SAS© library. The program will look for this connection in order to read frequency data from a SAS© dataset.

5. Running the program

5.1 Questionnaire method

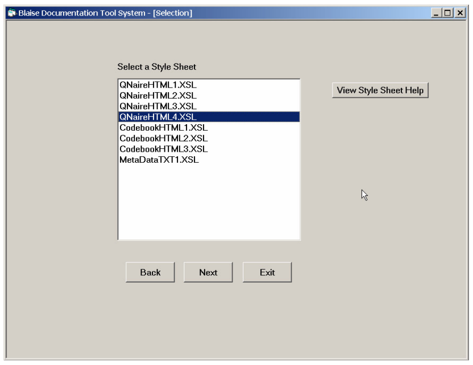
Once the program is installed, run BlaiseDoc via the normal methods. It will display a splash screen, followed by the first screen of the project wizard.

Version 0.4.15
Welcome to Blaisedoc
Blaise Documentation Tool
Press any key
Survey Research Center - Institute for Social Research
©2004 The Regents of the University of Michigan


Blaise Documentation Tool System - [Project Output Directory Selection]

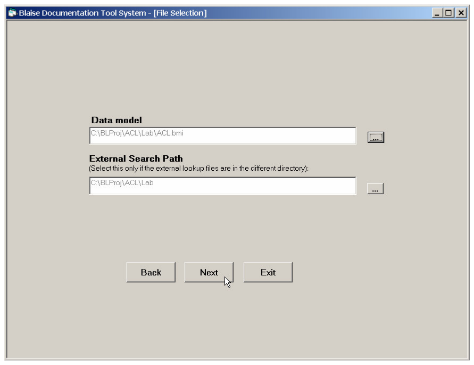
Output directory:

The Output directory is the location where the generated files (XML and HTML/RTF/TXT) will be stored. The datamodel and its external Blaise databases are next chosen.


Blaise Documentation Tool System - [Selection]

Select a Style Sheet

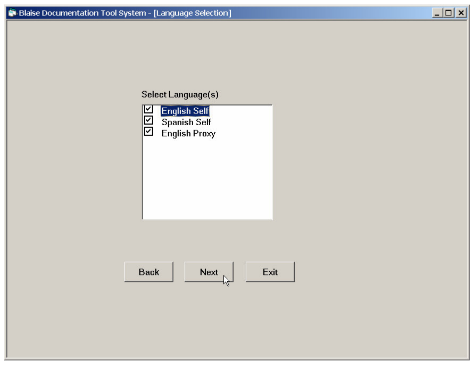
QNaireHTML1.XSL
QNaireHTML2.XSL
QNaireHTML3.XSL
QNaireHTML4.XSL
CodebookHTML1.XSL
CodebookHTML2.XSL
CodebookHTML3.XSL
MetaDataTXT1.XSL


Blaise Documentation Tool System - [File Selection]

Data model: C:\BLProj\ACL\Lab\ACL.bmi

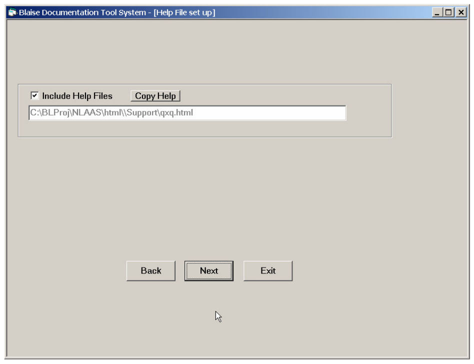
External Search Path
(Select this only if the external lookup files are in the different directory):
C:\BLProj\ACL\Lab

Next, select the languages to process. These are taken directly from the datamodel's defined languages. If a support database has been defined and the stylesheet that was selected indicates it, you will be asked to supply a properly marked help file that will annotate the output.


Blaise Documentation Tool System - [Language Selection]

Select Language(s)

☑ English Self
☑ Spanish Self
☑ English Proxy

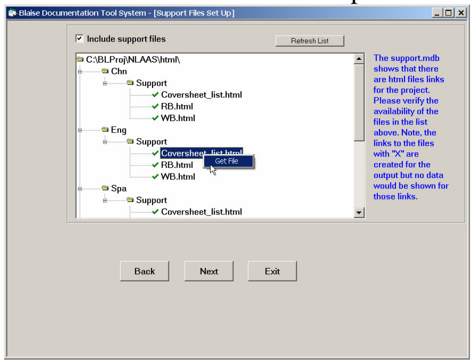

Blaise Documentation Tool System - [Help File set up]

☑ Include Help Files

C:\BLProj\NLAAS\html\\Support\qxq.html

If there are respondent booklet options stored in the support database then BlaiseDoc will present a map of the project and the support files it is expecting. Right-click and choose "Get File" to copy the appropriate html file to your project (output) directory.

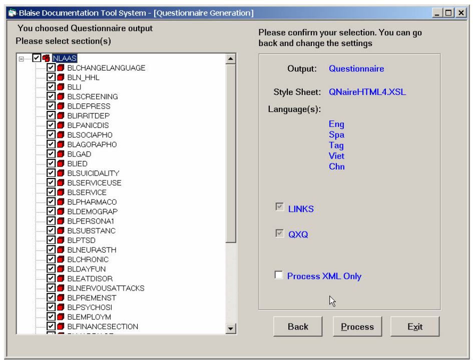
Finally, the program will ask you to select the blocks of data to process. The entire datamodel will be scanned so that fills and gotos are resolved, but only those blocks chosen will be part of the XML file and hence the output.


Blaise Documentation Tool System - [Support Files Set Up]

☑ Include support files

C:\BLProj\NLAAS\html\
- Chn
  - Support
    - ✓ Coversheet_list.html
    - ✓ RB.html
    - ✓ WB.html
- Eng
  - Support
    - ✓ Coversheet_list.html
    - ✓ RB.html
    - ✓ WB.html
- Spa
  - Support
    - ✓ Coversheet_list.html

The support.mdb shows that there are html files links for the project. Please verify the availability of the files in the list above. Note, the links to the files with "X" are created for the output but no data would be shown for those links.


Blaise Documentation Tool System - [Questionnaire Generation]

You choosed Questionnaire output
Please select section(s)

☑ NLAAS
- ☑ BLCHANGELANGUAGE
- ☑ BLN_HHL
- ☑ BLLI
- ☑ BLSCREENING
- ☑ BLDEPRESS
- ☑ BLIRRITDEP
- ☑ BLPANICDIS
- ☑ BLSOCIAPHO
- ☑ BLAGORAPHO
- ☑ BLGAD
- ☑ BLIED
- ☑ BLSUICIDALITY
- ☑ BLSERVICEUSE
- ☑ BLSERVICE
- ☑ BLPHARMACO
- ☑ BLDEMOGRAP
- ☑ BLPERSONA1
- ☑ BLSUBSTANC
- ☑ BLPTSD
- ☑ BLNEURASTH
- ☑ BLCHRONIC
- ☑ BLDAYFUN
- ☑ BLEATDISOR
- ☑ BLNERVOUSATTACKS
- ☑ BLPREMENST
- ☑ BLPSYCHOSI
- ☑ BLEMPLOYM
- ☑ BLFINANCESECTION

Please confirm your selection. You can go back and change the settings

Output: Questionnaire

Style Sheet: QNaireHTML4.XSL

Language(s): Eng, Spa, Tag, Viet, Chn

☑ LINKS

☑ QXQ

☐ Process XML Only

If the "Process XML Only" option is chosen it will run the stylesheet transformation against the current XML files. Transformation of just the chosen blocks is a future feature.

5.2 Codebook method

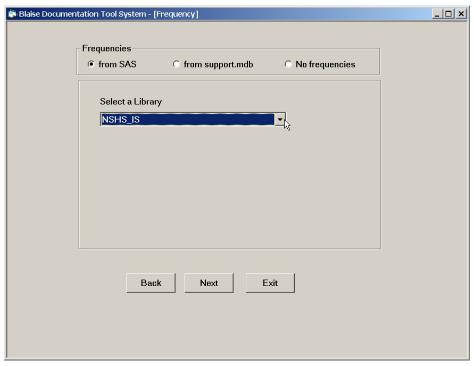
The process for creating a codebook is the same as a questionnaire except that there are additional options for the user. If SAS© has been installed on the system and the codebook stylesheet is one that uses frequencies then SAS© will start up.

SAS ODBC Driver

Waiting up to 60 seconds for the SAS System to start...

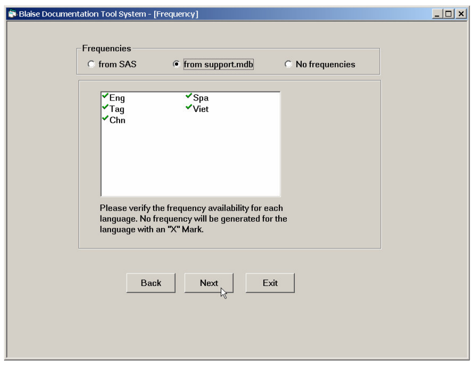
The user then has the option of choosing frequencies from SAS©, from the support database (to be discontinued), and no frequencies. The support database frequencies are stored in Access and simply report which language frequencies will be generated based upon the earlier language selection.

The SAS© frequencies are taken from the selected ODBC library.


Blaise Documentation Tool System - [Frequency]

Frequencies: [✓ from SAS] from support.mdb No frequencies

Select a Library: NSHS_IS


Blaise Documentation Tool System - [Frequency]

Frequencies: from SAS [✓ from support.mdb] No frequencies

✓Eng ✓Spa
✓Tag ✓Viet
✓Chn

Please verify the frequency availability for each language. No frequency will be generated for the language with an "X" Mark.

Note: if more than one language is available then a combined "all" language is assumed. BlaiseDoc expects frequencies for each language, including an "all." If there are no frequencies for a language or a particular field, then no frequencies for that question(s) will be shown.

6. Example output

The following examples are from the National Latino and Asian American Study, "all" languages.

6.1 Codebook (html)

Note the fill "this experience / these experiences" in brackets, the help reference "QxQ" and the "Respondent's Booklet" reference. The QxQ and RB references work the same across codebooks and questionnaires. The user can click on hot link to open another html document with the information about the question.

DS3

☞ (RB, PG 53)

What do you think was the <u>main</u> reason for [this experience / these experiences]? Would you say?

○	1 YOUR ANCESTRY OR NATIONAL ORIGIN OR ETHNICITY
○	2 YOUR GENDER OR SEX
○	3 YOUR RACE
○	4 YOUR AGE
○	5 YOUR HEIGHT
○	6 YOUR SKIN COLOR
○	7 YOUR SEXUAL ORIENTATION
○	8 YOUR WEIGHT
○	9 YOUR INCOME OR EDUCATIONAL LEVEL
○	10 OTHER

98 DON'T KNOW

99 REFUSED

View this question in...

External Links
QxQ | Respondent's Booklet

Universe

BLSCREENING.SC19=C01

DS2=C02

BLDS.DS3

Value	Label	Frequency	Percent	Cumulative Frequency	Cumulative Percent
.	MISSING	1437			
D	DON'T KNOW	149			
R	REFUSED	13			
1	YOUR ANCESTRY OR NAT	836	25.60%	836	25.60%
2	YOUR GENDER OR SEX	167	5.11%	1003	30.72%
3	YOUR RACE	711	21.78%	1714	52.50%
4	YOUR AGE	218	6.68%	1932	59.17%
5	YOUR HEIGHT	37	1.13%	1969	60.31%
6	YOUR SKIN COLOR	136	4.17%	2105	64.47%
7	YOUR SEXUAL ORIENTAT	16	0.49%	2121	64.96%
8	YOUR WEIGHT	66	2.02%	2187	66.98%
9	YOUR INCOME OR EDUCA	294	9.00%	2481	75.99%
10	OTHER	784	24.01%	3265	100.00%

- **Position:** 4581
- **Blaise Type:** Enumerated
- **SAS© Type:** Numeric
- **Decimals:** 0
- **Minimum:** 1
- **Maximum:** 10
- **Missing Data Codes:** ., .D, .R
- **Empty:** N

6.1.1 Codebook (html) – QxQ

The following is output from clicking on the QxQ (Question-by-Question objective or online help) link.

DS3

The Respondent is asked to identify the main reason for which s/he believes s/he had experiences related to discrimination. Record all the reasons mentioned. If the Respondent gives another reason that was not on the list, be sure to record it verbatim.

6.1.2 Codebook (html) - RB

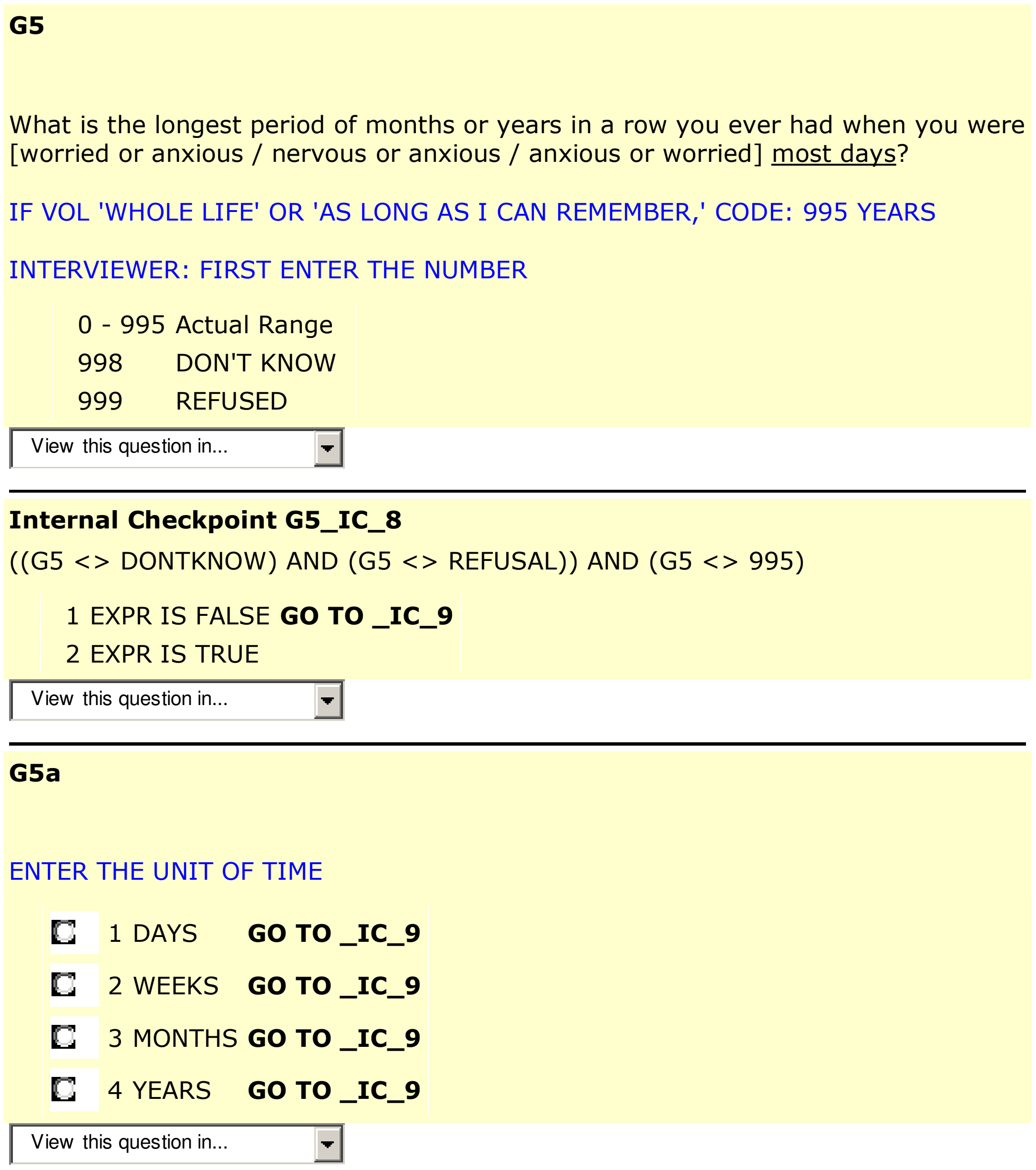
The following is output from clicking on the RB (Respondent Booklet) link.

WHAT DO YOU THINK WAS THE MAIN
REASON FOR THIS/THESE EXPERIENCE(S)?
WOULD YOU SAY . . . ?

- Your ancestry or national origin or ethnicity
- Your gender or sex
- Your race
- Your age
- Your height
- Your skin color
- Your sexual orientation
- Your weight
- Your income or educational level
- Other (Specify) _______________________

6.2 Questionnaire (html)

This sample question demonstrates the added DK/RF attributes on a range question. Note the numerical range programmed in the instrument is actually 0 through 97, but the allowable responses are 0 through 10, 97, DK, RF. Anything that is marked in magenta is typically hot linked to the appropriate item. For example, clicking on the GO TO _IC_9 reference will take the user to _IC_9 internal checkpoint in the current document.



G5

What is the longest period of months or years in a row you ever had when you were [worried or anxious / nervous or anxious / anxious or worried] most days?

IF VOL 'WHOLE LIFE' OR 'AS LONG AS I CAN REMEMBER,' CODE: 995 YEARS

INTERVIEWER: FIRST ENTER THE NUMBER

0 - 995	Actual Range
998	DON'T KNOW
999	REFUSED

View this question in...

Internal Checkpoint G5_IC_8

((G5 <> DONTKNOW) AND (G5 <> REFUSAL)) AND (G5 <> 995)

1 EXPR IS FALSE **GO TO _IC_9**

2 EXPR IS TRUE

View this question in...

G5a

ENTER THE UNIT OF TIME

○	1 DAYS	**GO TO _IC_9**
○	2 WEEKS	**GO TO _IC_9**
○	3 MONTHS	**GO TO _IC_9**
○	4 YEARS	**GO TO _IC_9**

View this question in...

G38d

☞ (RB, PG 9)

(Using a 0 to 10 scale on page 9 of your booklet, where 0 means no interference and 10 means very severe interference, think about the month or longer in the past 12 months when your [worry or anxiety / nervousness or anxiety / anxiety or worry] was most severe. What number describes how much your [worry or anxiety / nervousness or anxiety / anxiety or worry] interfered with each of the following activities during that month or longer?)

Your social life?

(IF NEC: How much did your [worry or anxiety / nervousness or anxiety / anxiety or worry] interfere with your social life during that time?)
(IF NEC: You can use any number between 0 and 10 to answer.)

IF 'DOES NOT APPLY',CODE 97

0 - 97	Actual Range	
98	DON'T KNOW	**GO TO G44**
99	REFUSED	**GO TO G44**

View this question in...

External Links
Respondent's Booklet

Consistency Checkpoint

Valid condition:	((((G38d >= 0) AND (G38d < 11)) OR (G38d = 97)) OR (G38d = DONTKNOW)) OR (G38d = REFUSAL)
Error returned to the user:	Error in G38d. Valid answers are 0 thru 10 and 97
Involved fields:	BLGAD.G38d

Constructed Variable G39

*INTERVIEWER CHECKPOINT (SEE *G38a - *G38d)*

1 ALL FOUR RESPONSES TO *G38a - *G38d SERIES EQUAL '0'
2 ALL OTHERS
8 DON'T KNOW
9 REFUSED

View this question in...

6.3 Questionnaire (rtf)

The following sample questions have been taken from a first draft RTF output using a training tutorial. Note that many features present in the HTML format are no longer available, such as drop down lists or hotlinks to other targets.

BAdvanced.MOD3_11
In the example question listed below, enter a 7 for "OTHER - SPECIFY" and type "I will be replacing my car with a motorcycle" in the specify window.

"If you were to replace your car, would it be with another car, some type of van, pickup, or utility vehicle, or what?"

1	**CAR**	**(GOTO BAdvanced.MOD3_13)**
2	**VAN**	**(GOTO BAdvanced.MOD3_13)**
3	**PICKUP**	**(GOTO BAdvanced.MOD3_13)**
4	**UTILITY VEHICLE**	**(GOTO BAdvanced.MOD3_13)**
7	**OTHER - (SPECIFY)**	
8	**DON'T KNOW**	**(GOTO BAdvanced.MOD3_13)**
9	**REFUSED**	**(GOTO BAdvanced.MOD3_13)**

Question Type: Enumeration

BAdvanced.MOD3_12
Please specify OTHER response.

Remember, to exit the response box, press [TAB] then [Enter], or press [ALT] and the "S" key together. To move to the next question afterwards, press [Enter].

DON'T KNOW Allowed
REFUSAL Allowed

Question Type: Memo

7. Technical problems encountered

BlaiseDoc was developed over a short time frame. Programming started in February 2004 and the first complete release was done July 2004. Not all development work went smoothly and there are certain surprises that appeared.

7.1 Slow navigation - RulesNavigator

A requirement of producing questionnaire documentation is to write out the questions in exactly the same order as presented to the interviewer. This bit of navigation was first done using the RulesNavigator within the BCP. This method, however, had several shortcomings that contributed to the slowness.

7.1.1 One RulesNavigator object per instance

Because of how BCP works with the datamodel, it usually is practical to only work with one copy of it in memory. Therefore all reference to the datamodel is done using pointers to the object, and any navigation via the RulesNavigator (perusing the rules) affects the content of all pointers. This would be similar to having a shared document; it gets updated and everyone sees the changes. This is usually not desirable when attempting to "hold" a place pointer, perform a look ahead, and then resume from the same spot.

Therefore it was necessary to have two copies of the same datamodel loaded into memory to accomplish this method.

7.1.2 Large overhead for loading the datamodel

As one can surmise, a large datamodel takes a chunk of memory and some time to load it. For every datamodel beyond the first additional system resources are required.

7.1.3 Inability to move to a field and navigate at random

The RulesNavigator method did not provide a method to move to a particular statement in the rules, or to pick up navigation from other methods (such as the Statements collection). It is purposely made to execute through all the rules of the datamodel from the first statement. In order to do a look ahead and return back to the same spot then required noting the statement the RulesNavigator started from, moving to a new destination, then start at the beginning of the rules and coming forwards again to resume.

7.1.4 Heavy recursion

The particular method used to navigate with the RulesNavigator made use of recursion. Unfortunately very complex instruments did force BlaiseDoc to run out of stack space (set by VB around one megabyte).

7.2 Memory use

BlaiseDoc is very string manipulation intensive and hence is very memory intensive. It relies upon aVisual Basics method to collect used strings and reallocate memory. However, Visual Basic hangs onto the memory used by a process until the process is released. This has the effect of slowly eating away the memory (although the process may currently be using a relatively small amount) until virtual memory comes into play (using disk space for memory). Once that happens processing speed dramatically slows down.

7.3 Not all information available - Procedures

BlaiseDoc has been limited by what is available via BCP (Blaise Component Pack). For example, any procedures written in the Blaise program are not accessible via BCC. For example, fills created by procedure calls cannot be fully resolved. Therefore, complete documentation is currently available for instruments without such calls.

7.4 Reusing variables for fills

As noted earlier, a common programming habit has been to reuse temporary variables for fills. This tends to lead to overly abundant text options when displaying fills on later questions. The problem still exists in the current release.

8. Programming Solutions & Suggestions

By far, the most impressive improvement in performance came from changing the method of rules navigation in the datamodel. The Statements collection provides nearly the same functionality as the RulesNavigator but does requires the programmer to manage the navigation through the instrument. The Statements collection offered the following advantages:

8.1 Simple loop navigation within minimum recursion

Instead of evaluating conditions to determine whether or not to go down an execution path, the Statements collection lists every statement within a block. The program simply loops through this list, and if it encounters a block it recurses to that block. This avoids the heavy recursion method using the RulesNavigator.

8.2 Can navigate freely

A particular statement can be reached quickly by using StatementIdent, an internal identifier to each statement in the Blaise datamodel. Using this method and the Statement navigation methods BlaiseDoc can freely and quickly examine statements to resolve gotos.

8.3 One datamodel only needed

Because quick navigation is now available, only one datamodel ever needs to be loaded. This reduces overhead greatly.

8.4 Low memory overhead

The routines that had once used pointers to the datamodel just pass a StatementIdent. With less recursion, one datamodel, and smaller parameters the memory use has been greatly reduced. However, the same problems for string manipulation note before are still present.

Given the limitations in the BlaiseDoc environment, the following suggestions are provided in order to help produce a more meaningful output.

8.5 Avoid procedures

Since BlaiseDoc uses the BCP, and BCP cannot read within procedures, place assignment of fills within auxblocks or within the code. This is obviously a less-than-optimal solution and is only suggested so that BlaiseDoc can fully quantify fills and goto statements.

8.6 Do not reuse fill variables

This should be taken only as a suggestion, but fills are not being resolved contextually within the rules section. So if fills are assigned multiple times in the same block the later questions will display too many options for that question.

9. Future enhancements

BlaiseDoc is still in its early stages and there is a list of enhancements waiting to be implemented.

9.1 Full meta information to be stored

If the goal is to reproduce the Blaise source code from the XML file then additional information needs to be stored. Currently only relevant information to producing documentation has been stored in the XML file. Certain categories of datamodel information have been left out, such as the layout, many modelib settings, and the datamodel properties.

9.2 Procedural information missing

BlaiseDoc has been limited by what is available via BCP (Blaise Component Pack). The current release of the BCP does not provide a method to retrieve the procedural information, and hence this limits the usefulness of some items (evaluating gotos and resolving fills).

9.3 Reading .HLP file directly

BlaiseDoc currently uses the original RTF file that has been annotated for use by the Microsoft HelpMake utility in order to merge the help file into the XML document. A future improvement will read the .hlp file directly since the source file is often not found with the datamodel.

9.4 Fills substitution library

In order to produce a readable document some fills need to be changed by hand. For example, ^xFill[1] may be correct, but if it is assigned within a procedure then BlaiseDoc cannot resolve it any further. Hence an external database that translates xFill[1] into "easy/fast" would be much more readable. This same library could be used for fill expressions such as "A[1] + ' and ' + SUBSTR(xLocation, 1, 5) + ' or that.'"

9.5 More context sensitive fill resolution

Fills that are reused between questions appear to have more values in later questions than in earlier ones. The reason behind this is they did indeed have all the values listed, but only a subset of those values are valid in the context of the question. The routine to determine fills needs to be more intelligent and determine only those values for a fill for the currently specified question.

That is, suppose before A1 the fill xFill was assigned either "one" or "two" based upon some count. Question A1 displays the fill, and BlaiseDoc will show the fill as "(one/two)." Then just after A1 the fill is reused and is assigned either "three" or "four" and used in A2. Question A2 will now show "(one/two/three/four)" instead of just "(three/four)."

9.6 Full evaluation of expressions

Because of the complexity involved in determining an expression for all its values, BlaiseDoc produces a number of "internal checkpoints." The output would be much tighter (especially more gotos via enumerated questions) if BlaiseDoc interpreted the following base expressions:

<
>
<=
>=
NOT
function calls
IN operator

9.7 Clear indication of arrays/question blocks

The current output does not show the start/end of arrays of blocks or the logical range for questions within a block. The latter may be of interest to programmers and data analysts.

9.8 Reprocessing Stylesheets

The "Process XML Only" option on the last screen of the BlaiseDoc wizard runs the current stylesheet transformation against the current XML files. An option for the future is to intelligently reprocess those blocks chosen so that processing of very large or complex datamodel can be done in steps.

10. Conclusion

BlaiseDoc is still early in its lifecycle but has proven to be a valuable tool for producing documentation from the Blaise system. We expect it will continue to improve and expand to its full potential.

If you have questions or comments, please contact Peter Sparks (zebulon@umich.edu) or Youhong Liu (yliu@umich.edu).