Web-based CAI System for Blaise Instruments Development

Lilia Filippenko and Sridevi Sattaluri, RTI International, USA

1. Introduction

Over the course of conducting data collection for many studies, RTI recognized the need to unify the instrument design and documentation process for all CAI studies regardless of the mode of data collection. To achieve this goal, RTI developed a full featured web-based CAI system, Hatteras™, which allows instrument designers and programmers to work concurrently and supports multi-mode data collection efforts for CAWI, CAPI, and CATI instruments.

Hatteras™ consists of three major components: SurveyEditor, SurveyEngine, and Commander. All specifications are stored in database tables through the user interface provided by SurveyEditor. For web-based instruments SurveyEngine reads the tables to produce the web pages. For Blaise instruments a client utility Hatteras™ Commander is used to generate Blaise source code and scripts for Audio Computer Assisted Self Interview (ACASI). Hatteras™ provides multiple language support. Translation specialists can edit and comment the instrument via the SurveyEditor. Through version control, Hatteras™ permits multiple tasks to be performed on the same instrument, allowing programmers, translators, reviewers, and the design team to work simultaneously. It keeps an audit trail or history of any change made to the instrument.

This paper describes in some details how Hatteras™ SurveyEditor and Hatteras™ Commander were adapted to speed-up development of multilingual Blaise instruments.

2. Background

Since 2004 the Questionnaire Specifications Database (QSD) was used at RTI to develop Blaise instruments. QSD consists of tiers for the user interface and business rules, and the back end relational database. Microsoft Access had been chosen at that time based on its simplicity, its familiarity among the programmers, and to ease the transition from working with specifications exclusively in word processing documents. In using QSD on a number of studies, we at RTI International have found that it reduces Blaise questionnaire development time especially those with a second language. But problems with using Access databases remotely and limitation creating only bilingual instruments pose a need for using a web interface and SQL Server as the back end.

We decided to use existing web-based CAI system Hatteras™ that is used for all web-based studies at RTI and was developed by RTI programmers. Hatteras™ is in production since 2006 and has since been utilized on several large scale survey projects. We looked at different features of Hatteras™ and found that many of them are very close to what we used to have:

- The Hatteras™ Commander has an option to upload a text file that conforms to the Hatteras™ spec format – it is the first step in getting specs in QSD.
- The Hatteras™ Survey editor website is where all instrument specifications are saved – it is sufficient to use for Blaise instruments as well.
- Comments are a way to communicate changes needed – they are compatible with feature in QSD to monitor problems.
- Specs can be outputted to a readable format

There are many additional features that are used for web instruments, but are not needed or cannot be used with Blaise instruments. The most important one of them is the Hatteras™ engine that creates pages on-the-fly to conduct web interview. For us, it meant that Hatteras™ needed to have an option to generate Blaise code and have a few additional settings to recognize what kind of instrument is loaded. Although Hatteras™ is a great system for work with the web instruments, we will describe in this paper only the features used to develop Blaise instruments.

3. Hatteras™ SurveyEditor

3.1 Overview

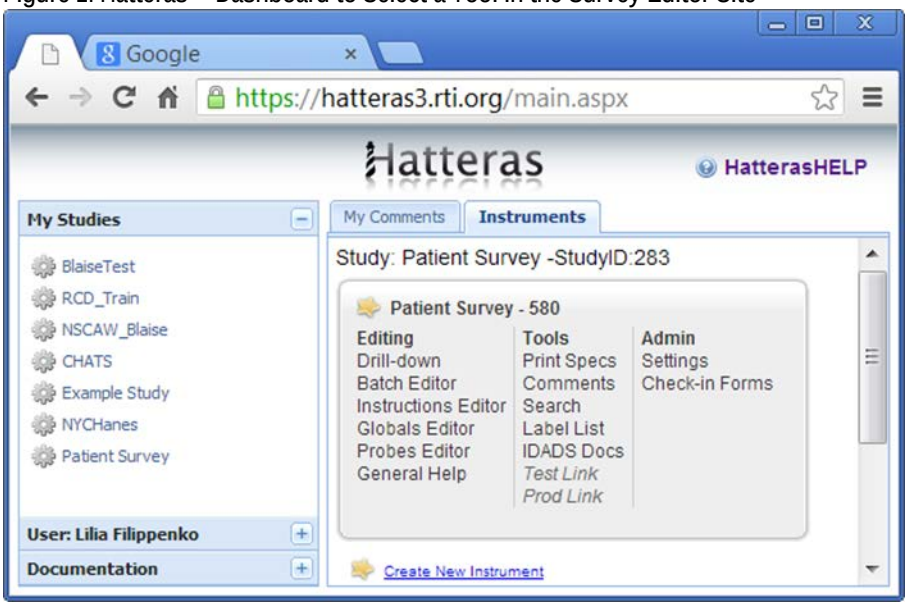
SurveyEditor is the interactive instrument development portion of Hatteras™. SurveyEditor provides the user interface for editing all aspects of a survey instrument including forms, items, comments, program logic, and other applicable information. Hatteras™ stores the meta-data for all instruments created using Hatteras™ in the SQL Server database.

A system admin is the only person that can setup a new user or change instrument assignments for a user. When new user is added to Hatteras™, admin will assign a user name for the user, enter their first and last names, enter their RTI email address, and assign a password for the user. These credentials will be used for all studies available for the user in Hatteras™. There are 5 Roles to use in the Hatteras™ Editor:

- Client - Read only. Client may only add comments to forms.
- Language Specialist / Spec Writer - Can change specs in the instrument, except for code blocks.
- Programmer - Can change all specs in the instrument.
- Study Admin - Can access the Admin tools available for the instrument.

These roles can be assigned for each instrument. Once the user has logged in, a list with available studies is shown on the left. When the user clicks on a study, the instruments for that study will show up on the right, under the Instruments tab. To start editing an instrument, the user clicks on the Drill-down editor for that instrument.

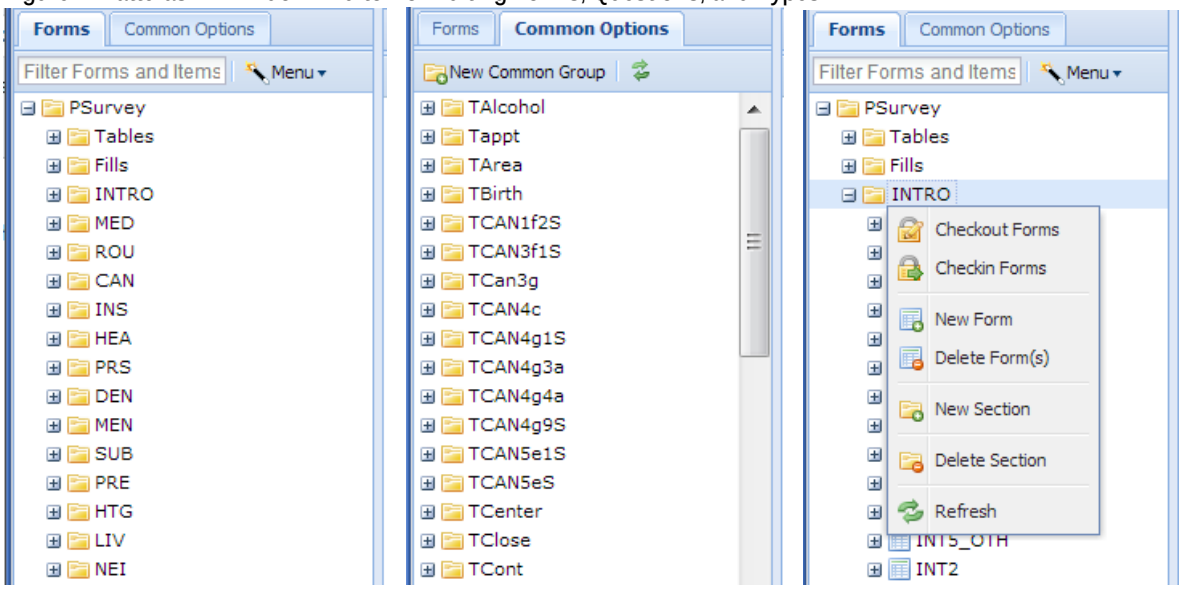
Figure 1. Hatteras™ Dashboard to Select a Tool in the Survey Editor Site



3.2 Drill-down Editor

To start editing an instrument the user clicks on the Drill-down editor for that instrument under “Editing” on the dashboard. The Drill-down editor is the primary instrument editing interface. The left menu displays all sections in the instrument under “Forms” tab and all enumerated types used in the instrument under “Common Options” tab.

Figure 2. Hatteras™ Drill-down Editor for Editing Forms, Questions, and Types



There are five levels of detail in an instrument:

Section – a grouping of forms

Form – the page which will be shown in a web instrument

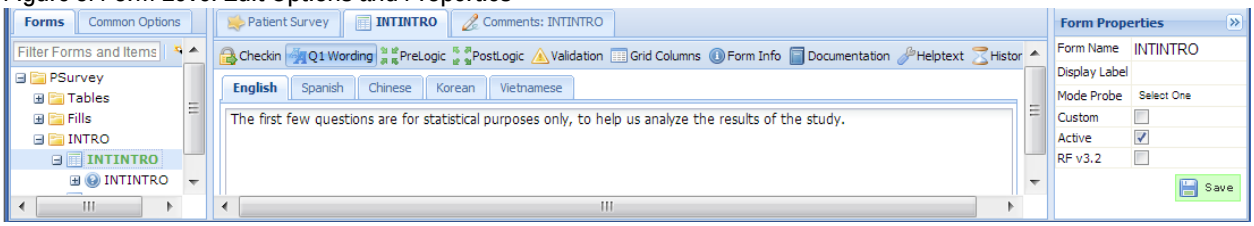
Question – the main question on the form

Item – the data element(s) collected on the form

Item option – properties of the item

To view a specific form, the user clicks on that form in the left menu. This will expand to display all questions and items contained in that form. In Blaise instruments each form can have only one question and usually one question has only one item. The item name is served as a field name in the Blaise instruments. When a question has a few items, the question wording is added as a stem to all items for this question.

Figure 3. Form Level Edit Options and Properties



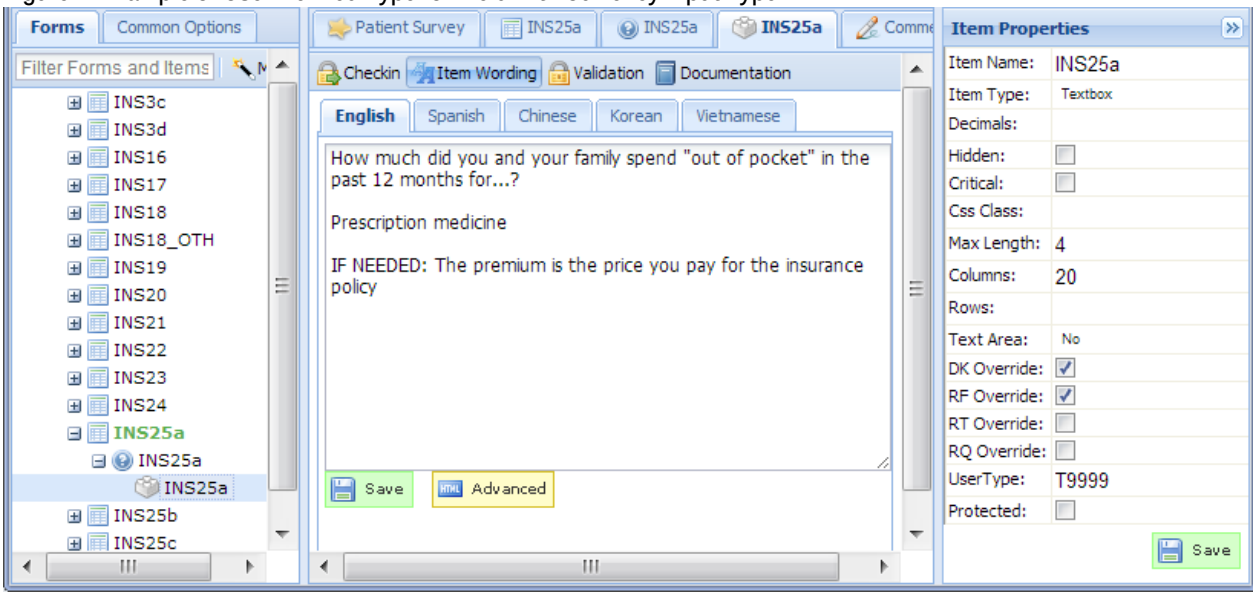
Different options are available at form, question, and item levels. Some of them are described in this paper.

3.3 Blaise Specific Options

Hatteras™ section, form, and question options are used for Blaise instruments development exactly in the same way as for web instruments. But at item level new properties were added and special rules for using existing properties were designed to accommodate specific Blaise instrument needs.

To use Blaise user-defined types in Hatteras™, property "UserType" was added. It is also used when a question or item type unique to Blaise is needed. For example, an array of textboxes can be specified as a simple looping section with one textbox type question but a simpler way is to just specify the "ARRAY [1..n] OF string[20]" in the "UserType" property of the item.

Figure 4. Example of User-Defined Type for Field with Currency Input Type



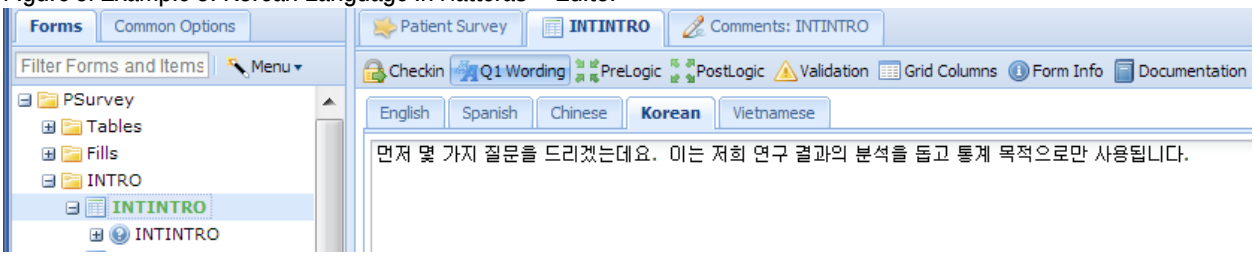
In addition, following rules are used for the Blaise instruments:

- Special Blaise field attributes "description" and "tag" can be specified on the "Documentation" tab for the item. The "Label" serves as the description, and the "Analytic" property is served as the tag for the Blaise field.
- To use a field with the "SET OF" type in Blaise, in Hatteras™ the question is defined as "check box" and its item is defined as "radio".
- Within the drill down editor, the instrument level setting can be overridden by setting the "DK Override", "RF Override" and "RQ override" properties of the item to not allow "Don't Know", "Refusal", and/or "Empty" respectively for the item.

3.4 Managing Languages

Hatteras™ allows language specialists to translate and edit the instrument in multiple languages. The translated text can initially be mass imported into Hatteras™, and edits can be made via the Hatteras™ Editor. The language specialist can view the "master" language (usually English) at the same time they are viewing and editing the alternate language(s).

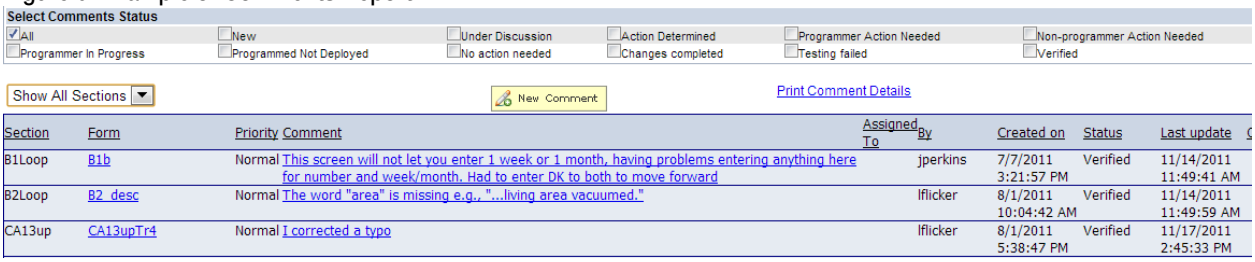
Figure 5. Example of Korean Language in Hatteras™ Editor



3.5 Comments and History

Hatteras™ allows structured, documented communication of specifications between programmers, specs writers, translators, and clients. It provides a robust comment feature, which allows all designers, testers, and programmers to log comments associated with a particular form. Comments provide a way to communicate changes needed and provide to-do lists based on comment statuses for instrument designers and programmers. Comments can be accessed two ways. The main way is through the comments link under Tools on the dashboard. Clicking on the "Comments" link from the dashboard takes the user to the comment report for that instrument. The other way is through the Drill-down editor.

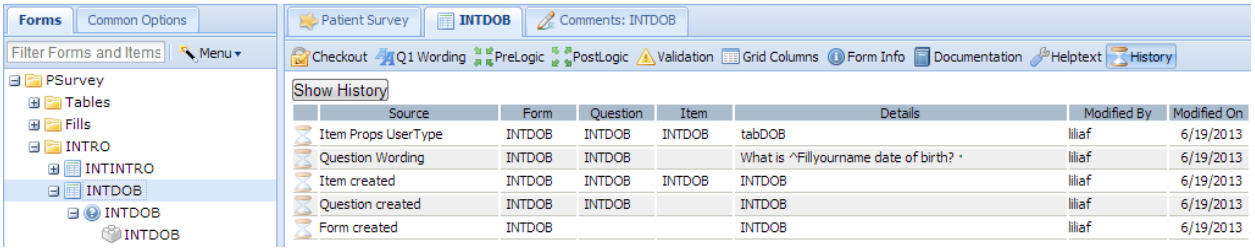
Figure 6. Example of Comments Report



Section	Form	Priority	Comment	Assigned To	By	Created on	Status	Last update
B1Loop	B1b	Normal	This screen will not let you enter 1 week or 1 month, having problems entering anything here for number and week/month. Had to enter DK to both to move forward		jperkins	7/7/2011 3:21:57 PM	Verified	11/14/2011 11:49:41 AM
B2Loop	B2_desc	Normal	The word "area" is missing e.g., "...living area vacuumed."		lflicker	8/1/2011 10:04:42 AM	Verified	11/14/2011 11:49:59 AM
CA13up	CA13upTr4	Normal	I corrected a typo		lflicker	8/1/2011 5:38:47 PM	Verified	11/17/2011 2:45:33 PM

Hatteras™ allows individuals to check-in/check-out forms which prevents duplication of effort. It keeps an audit trail of changes to each form in the instrument. History tab in the Drill-down editor provides a detailed record of all changes made to the form. Clicking the "Show History" button will display all changes.

Figure 7. Example of Form History Changes



Source	Form	Question	Item	Details	Modified By	Modified On
Item Props UserType	INTDOB	INTDOB	INTDOB	tabDOB	liliaf	6/19/2013
Question Wording	INTDOB	INTDOB		What is ^Fillyourname date of birth? ·	liliaf	6/19/2013
Item created	INTDOB	INTDOB	INTDOB	INTDOB	liliaf	6/19/2013
Question created	INTDOB	INTDOB		INTDOB	liliaf	6/19/2013
Form created	INTDOB			INTDOB	liliaf	6/19/2013

4. Hatteras™ Commander

4.1 Overview

The Hatteras™ Commander is the client utility used to assist with testing, debugging and other tasks needed for Hatteras™ Instruments. When the user starts it, the credentials as the one used in the

Hatteras™ Editor website are entered to login. After logging in, the user selects one of the available Study/Instrument to see more tabs. Only a few of these tabs are used by the Blaise developers.

Figure 8. Hatteras™ Commander Main Screen

The “Multi-language” tab is used to access the multi-language features of Hatteras™. Languages used for the study are checked in the list of supported by Hatteras™ languages and the “Master language” is selected to define the default. Once the user has setup supported languages, they are available in the Hatteras™ Editor website for translators. In addition, the “Multi-language” tab is used to do bulk exporting and importing of strings for translation.

The “Spec Loader” tab is used to upload a text file that conforms to the Hatteras™ spec format. Once uploaded, an instrument programmers, spec writers, and translators can modify the instrument from the Hatteras™ Editor website.

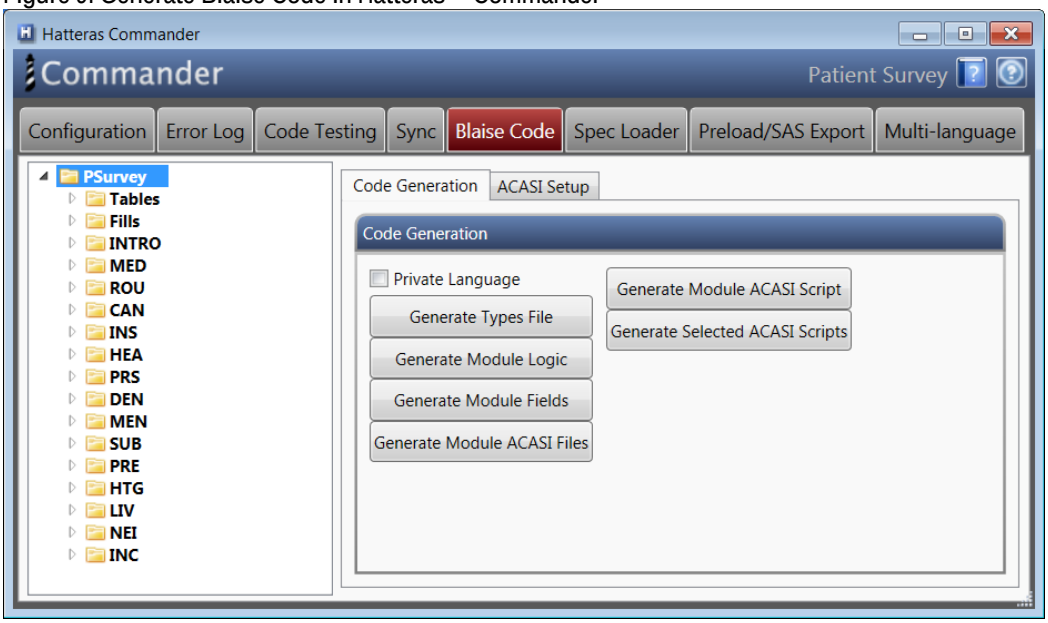
4.2 Generating of Blaise Code

The “Blaise Code” tab is used to generate Blaise files from the currently selected instrument. Each section in our Blaise instrument is programmed as a separate block with additional sub blocks as desired. All enumerated types are defined at the instrument level and are in one Types file. Since Blaise allows the separation of fields and rules at the block level as well, each block has an INCLUDE file which contains all fields associated with the specific block. This important feature of Blaise allows dividing responsibilities between programmers, specs writers, and translators.

To generate Types file for the instrument, the user clicks on the "Generate Types Files". A dialog appears asking the user to enter a location and file name. Once they are selected, a progress bar appears informing user of the file generation status.

To generate Blaise Logic or Fields file for a section, the user selects the section in the list and then clicks on the "Generate Module Logic" or the "Generate Module Field" button respectively. The file names will be pre-fixed with the name of the section. It is possible to generate files for all sections at one; to do so the user clicks on the section at the root level.

Figure 9. Generate Blaise Code in Hatteras™ Commander



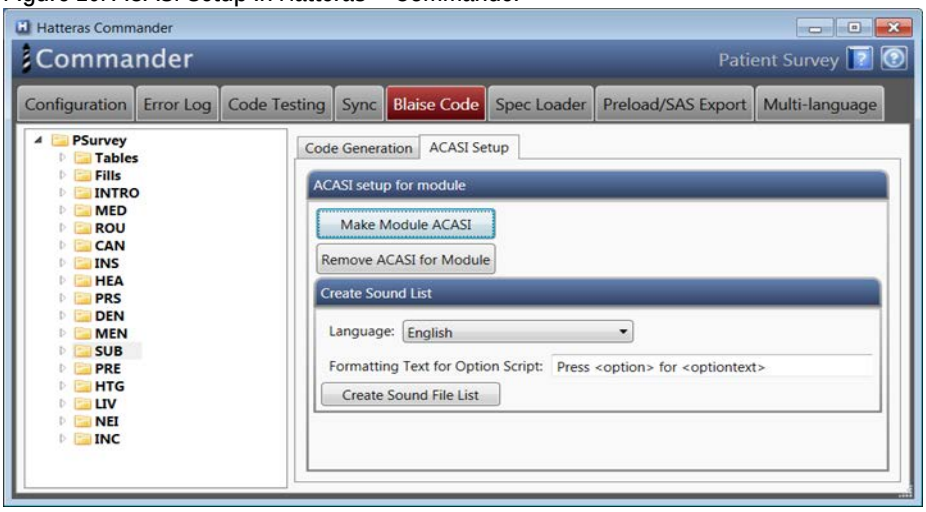
Programmers work in the Logic files only. Fields and Types files are created after changes are completed in the Hatteras™ Editor by specs writers and translators. The programmer's goal is to use an approach where there is no hard coded text in the Blaise instrument. This is especially important for the Blaise instruments with Non-Western languages when Blaise Control Center cannot be used to edit text in such languages. All files are created as UTF-8 with a BOM (Byte Order Mark).

4.3 Setup ACASI with Blaise Instruments

For studies that require use of audio files, Hatteras™ helps to produce scripts for recording and generate Blaise code to play them during an interview. To set up a module as an ACASI module and manage the sound files associated with the questions in the module, user uses the "ACASI Setup" tab in Hatteras™ Commander.

After selecting the module, user clicks on the "Make Module ACASI" button to associate a tag that identifies the questions as an ACASI items.

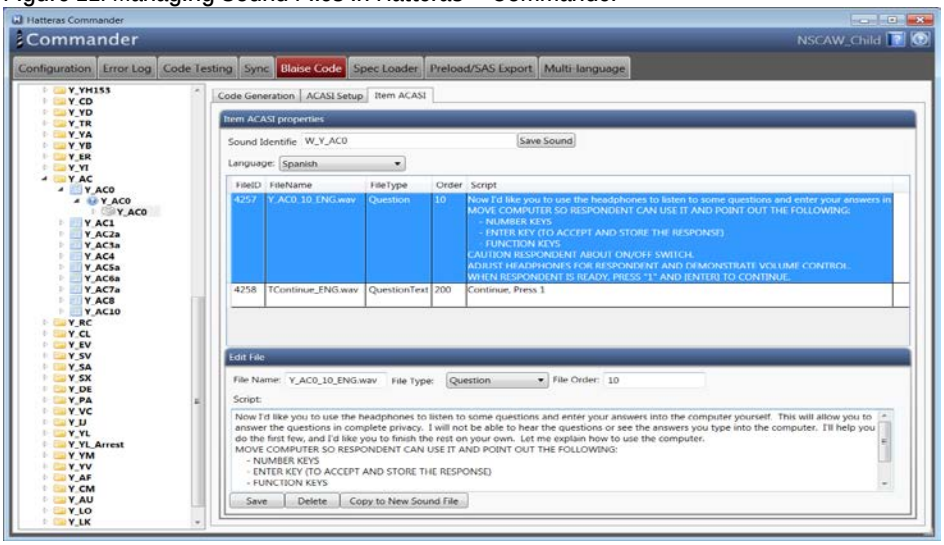
Figure 10. ACASI Setup in Hatteras™ Commander



In the "Create Sound List" panel, user selects the language and clicks the "Create Sound File List." This will create the scripts for the questions and for the response options text for each question in selected language. The formatting text for the options script can be modified as required. With the default text, the option text script for as Yes/No question would be generated as "Press 1 for Yes; Press 2 for No."

By selecting each question's item, user can view the item level ACASI properties. A new tab "Item ACASI" appears. This tab provides the interface for editing an existing script, adding a new script file and deleting existing ones.

Figure 11. Managing Sound Files in Hatteras™ Commander



In the Blaise instrument "Sound Identifier" is used as an "AUXFIELD" to hold information about all sound files associated with the question. The list of sound files for a question can be modified and text for scripts can be edited in the "Edit File" panel.

The "Generate Module ACASI files" on "Code Generation" tab is used to generate relevant ACASI code files used in the Blaise instrument. The "Generate ACASI Scripts" and "Generate selected ACASI scripts" are used to generate word documents containing scripts for the recording artiste.

5. Summary

Hatteras™ is the software used at RTI International for developing the web-based instruments during multiple years. With addition of Blaise Code Generation feature, Hatteras™ became the standard system used by all questionnaire developers at RTI International regardless what programming language is required for the study.

Features of Hatteras™ used for the Blaise instruments development include:

- **Enhanced Workflow Process** – Hatteras™ allows structured, documented communication of specifications between programmers, designers, and clients. It keeps an audit trail and history of changes to each form in the instrument.
- **Multiple Language Support** – Hatteras™ allows translation specialists to edit, comment, and test the instrument in multiple languages.
- **Output of Programming Code** – Hatteras™ outputs the complete programming code for items of simple structure without logic branching. The output code serves as a starting point for more complex programming.
- **Import Specifications in Multiple Formats** – Hatteras™ can import specification files from word documents formatted according to a pre-defined template.
- **Output Specifications** – Hatteras™ displays and outputs interview sections in several formats, depending on the needs of the reviewer.
- **Output Codebooks** – Hatteras™ stores all metadata about the instruments. The metadata is used to generate the codebooks with the original questions, variable names, and response frequencies.

Using the unified web-based Hatteras™ system across RTI International allows programmers, specs writers, and language specialists simplify training process and speed up development of multi-mode and multilingual CAWI, CAPI, and CATI instruments.