

Multilingual Survey Development in Blaise: Streamlining Localization

Karl Dinkelmann & Kelly Lieske; University of Michigan - Survey Research Center

1. Introduction

Surveys targeting populations across linguistic boundaries must consider translating survey content into the languages spoken by respondents. Doing so ensures inclusivity and accessibility, reducing barriers to participation for diverse communities. Many surveys conducted in the United States originate in English, the source language, and are subsequently translated into Spanish, the target language. However, when a target population does not possess literacy in English/Spanish or the original survey language, it is essential to expand the range of languages offered. This consideration becomes increasingly important in cross-cultural research settings and when studying minority populations.

This multilingual approach may take various developmental forms, but translation remains central. Harkness et al. (2010), in the "Translation" chapter of the Guidelines for Best Practice in Cross-Cultural Surveys, define translation as the "process of expressing the sense of words or phrases from one language into another," also known as "Asking the Same Questions and Translating (ASQT)". They further clarify: "Translation procedures are critical in multinational, multicultural, or multiregional surveys, commonly called '3MC' surveys. Although high-quality translations alone do not guarantee survey success, poor-quality translations can severely hinder a project's effectiveness by obstructing the collection of comparable data."

Savourel (2001) notes that localization involves adjusting a product to be linguistically and culturally appropriate for different target audiences. Localization is "adapting a product's translation to a specific country or region". It constitutes the second phase in a broader translation and cultural adaptation process, Internationalization and Localization. Unlike simple translation, language localization entails comprehensively analyzing the target culture to adapt products accurately to local needs ("Language localisation," 2025). Multilingual surveys vary significantly in scope; governments mandate some to include specific languages, while others include languages necessary to represent their targeted populations effectively.

Survey authoring applications offer features to create and administer multilingual surveys, a notable strength of the Blaise platform. Blaise 4 supported multilingual survey creation but had a significant drawback: its underlying architecture and encoding were based on ASCII. In contrast, Blaise 5 overcomes this limitation with an architecture that natively supports Unicode encoding, capable of representing all modern and complex language scripts.

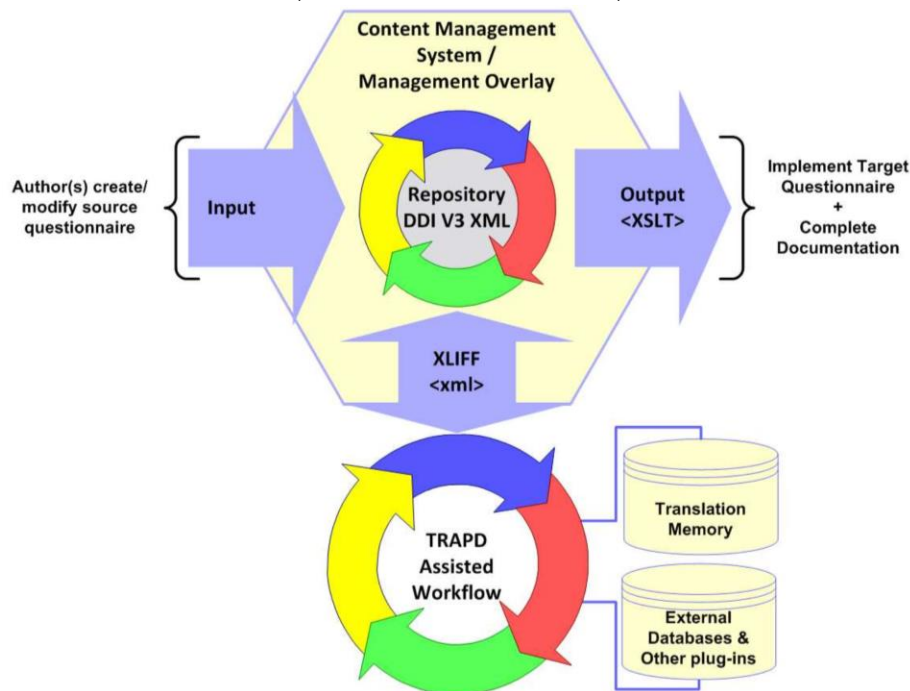
This paper examines strategies for streamlining and automating translation workflows of Blaise source code, aiming to enhance the efficiency of translating and localizing survey content. We outline the current challenges and review relevant components from the perspective of translation researchers. Additionally, we explore the advantages and disadvantages of utilizing Blaise's out-of-the-box translation tool, specifically the Data Model Texts Editor and BITT file, while considering supplemental parsing techniques to optimize the presentation of survey text to translators, typically non-developers. We aim to

identify methods for systematically extracting text from Blaise source code, facilitating an effective translation process, and establishing mechanisms for reintegrating translations back into the source code. Moreover, we discuss integrating these processes with the XML Localization Interchange File Format (XLIFF) standard, aligning our workflow with industry-standard localization tools.

2. Background

Conducting survey translations is more rigorous than many other translation projects. The origin of our process comes from work done two decades ago, presented as a blueprint for a translation tool (Harkness, Dinkelmann, Pennel, & Mohler, 2007). This blueprint proposes developing and specifying an assisted translation workflow designed to improve translation efficiency, quality, and documentation for cross-national survey research, specifically the European Social Survey (ESS). The proposed process follows the systematic translation process called TRAPD (Translation, Review, Adjudication, Pre-testing, Documentation) (Harkness, 2010). The report reviews technical solutions designed to assist with the complexities of the translation workflow, suggesting using eXtended Markup Language (XML) standards to help with the storage and translation movement between translators and questionnaire authors. It proposes to use the Data Documentation Initiative (DDI) standard as the central repository for the survey questions and parse out the translatable texts to store them in the localization industry standards of XLIFF, allowing the workflow to tap into freely available XLIFF-aware open-source software to translate surveys. An overview of this blueprint workflow, which Harkness et al. (2007) proposed, is shown in Figure 2.1 below.

Figure 2.1: Overview of Blueprint Workflow
(taken from Harkness et al., 2007)



The proposed tool was designed to facilitate simultaneous translation and collaborative review of survey questionnaires across multiple languages while enhancing process control, version management,

documentation, and overall translation quality. Additionally, it aimed to leverage advancements in localization technologies and electronic collaboration tools to streamline workflow efficiency. Figure 2.2 below illustrates the TRAPD process as detailed in the blueprint.

Figure 2.2: TRAPD process (taken from Harkness et al., 2007)

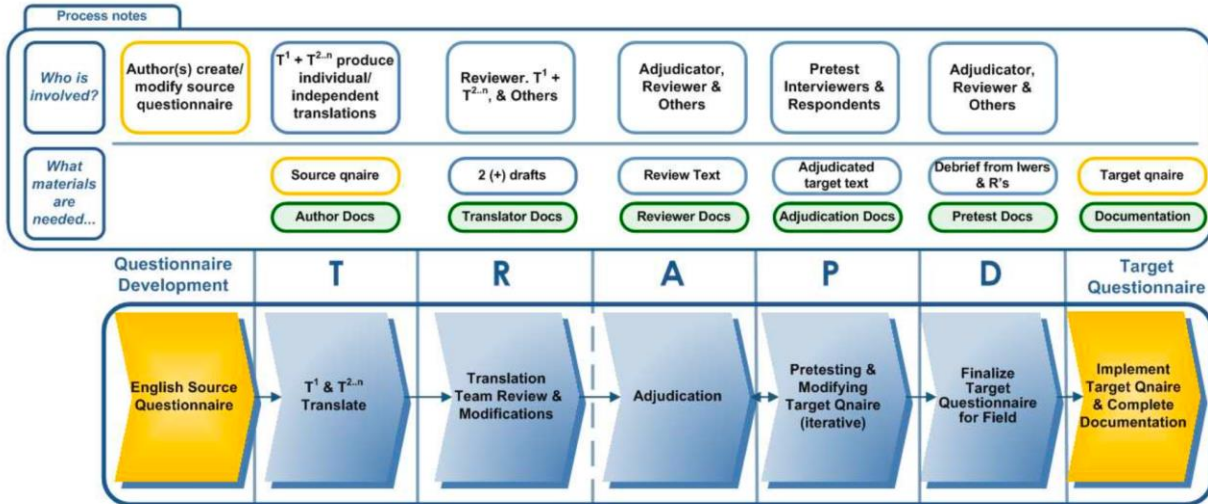


Figure 2.3 below shows the TRAPD-Assisted Workflow and maps out the overall process proposed in the blueprint, and Figure 2.4 describes the summary of the tasks called out in the TRAPD-Assisted Workflow below.

Figure 2.3: TRAPD-Assisted Workflow (taken from Harkness et al., 2007)

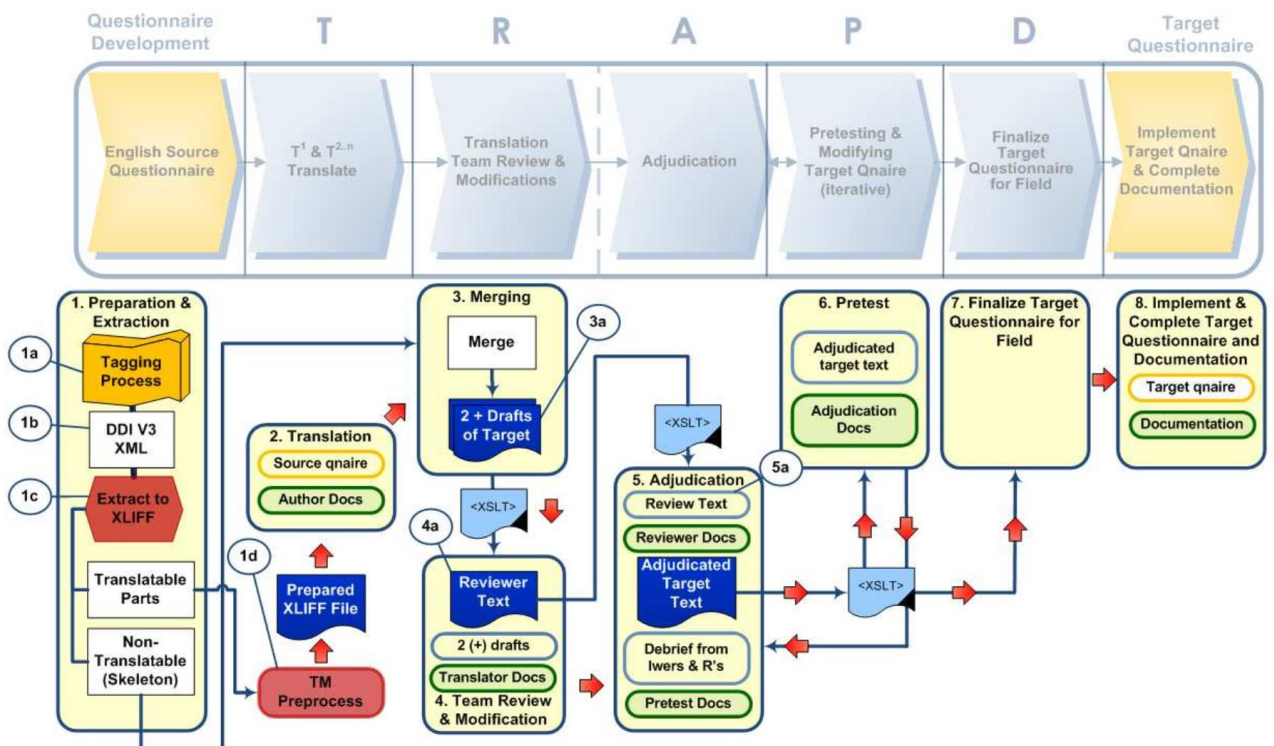


Figure 2.4: TRAPD-Assisted Workflow – Task Summary

(taken from Harkness et al., 2007)

Step	Task	Task Summary	Next Step
1a	Source Markup/Tagging Process	A tool that assists in the markup process: automated, using a series of MS Word macros; parsing elements to the appropriate locations within the new data collection schema (elements and attributes).	2
1b	Structured (DDI 3.0 XML) File	Parsed DDI compliant (V3) XML File (or instance): 1. Could have been recently marked up for translation 2. Obtained from a data archive 3. Previous wave, etc.	
1c	Extraction	Extract DDI XML text to XLIFF file format. As part of the standard, the extraction process creates a “skeleton” file to allow for the translated content to merge properly.	
1d	TM Preprocess	Preprocess the XLIFF file with translation memory text	
2	Translation #1 Translation #2...n	Translate source content: Translators uses a XLIFF-aware tool to translate content, see author annotations, and document translation process.	3
3	Merge	The translated content is merged back into its original file format (a DDI 3.0 XML file). XSLT template is used to merge the appropriate information forward in the lifecycle (i.e. documentation and other metadata elements that are important for future translations as well as secondary analysis of the resulting data from that survey).	4
3a	Draft Target XML file	Initial draft target DDI 3.0 XML; one file per translator.	
4	Team Review	Markup/review tool: Combination tool merging the items from the translation phase, with various display options. Allowing one to combine, review, refine, and comment.	5
4a	Team Review Results	The new DDI 3.0 file is then created.	
5	Adjudication	All items from the previous steps brought forward.	6
5a	Adjudication Results	Modified DDI 3.0 file.	
6	Pretest	Tool allowing one to output to a pseudo-PAPI instrument, or a file that is easily parsed into a CAPI instrument.	5 or 7
6a	Pretest Results	Outcomes move to adjudication for next steps.	
7	Finalize Target Questionnaire for Field	Each country would finalize their instrument, including adding country or organization-specific information.	8
8	Target Questionnaire Created and Implemented	DDI XML file to be passed forward in the life cycle, with documentation included in the file. Publishing files: 1. Target Questionnaires and documentation. 2. Publishing the files for Pretest (several different outputs necessary here, html, rtf version, and possibly a version that assists in inserting the text into a CAI application). 3. Locale-specific output via XSLT templates.	(1) or Done

In retrospect, this TRAPD-Assisted Workflow proved overly elaborate, introducing additional complexities to an already intricate process.

3. The problem

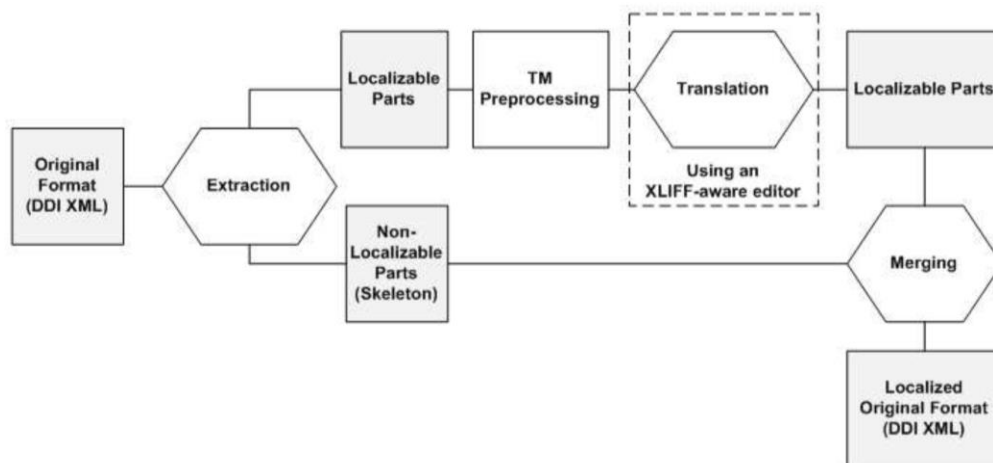
Adapting additional language translations to Blaise surveys introduces significant challenges, particularly as the number of languages increases. While the scope of translation addition increases linearly, combining and managing all languages within the same data model can become unwieldy at an exponential pace. Manually extracting, translating, and reintegrating survey text (including response categories and on-screen interviewer instructions) into the Blaise source code is labor-intensive and inefficient, involving question-by-question cutting and pasting one language at a time.

Automation will minimize repetitive tasks, allowing instrument programmers to focus on complex, high-value components of survey development. The resulting efficiency gains aim to deliver substantial time savings, reduce errors, and enhance the scalability of multilingual survey creation. The resulting process will establish a streamlined and adaptable solution for integrating additional languages into Blaise surveys, setting a new standard for multilingual survey development.

The work that was part of the specification for the TRAPD-Assisted Workflow blueprint was too complex. Using DDI as the survey repository is good in theory, but for our purposes, we have the Blaise source code that needs to be translated and transformed to a standard that allows us to follow an extract-translate-merge process (Savourel, 2001) that meets our needs most of the time. Figure 3.1 displays the principles of text extraction using XLIFF and how data flows through an extract-translate-merge model (Savourel, 2001).

Figure 3.1: Principles of text extraction using XLIFF

(taken from Harkness et al., 2007)



With complex multilingual projects on our horizon at the University of Michigan, one with eight languages and another with sixteen, we need a better way of managing translations within Blaise. However, as previously noted, the TRAPD-Assisted Workflow was too complex. Nevertheless, the directions discussed in the blueprint (Harkness, 2007) could be helpful in a new but simplified process.

4. Blaise: Out-of-the-box solution

Blaise allows users to extract translatable survey text within a Blaise solution to a BITT file. The Blaiswe online help file (5.15.2, 2015) describes it this way:

Figure 4.1: Blaise translation process

Run the Data model Text Editor



- Double click in the [Solution Explorer](#) on the project's bitt-file to open the [Data model Text Editor](#). The translation file contains all texts per mode, role & language found in the following sections: SPECIALANSWERS, MODES, TYPE, CONST, (AUX)FIELDS, BLOCK, GROUP, CHECKS. Enumerations use category names as label text if the text for a category label is missing. These text specifications from the bitt-file are then used during compilation of the data model. Note that texts in the rules are not included. Consider whether text assigned to a variable in the rules can be placed in the CONST section and text defined on a check in the rules can be moved to a CHECKS section.
- Alternatively you can run the '*DataModelTextsEditor.exe*' in the Blaise installation folder.

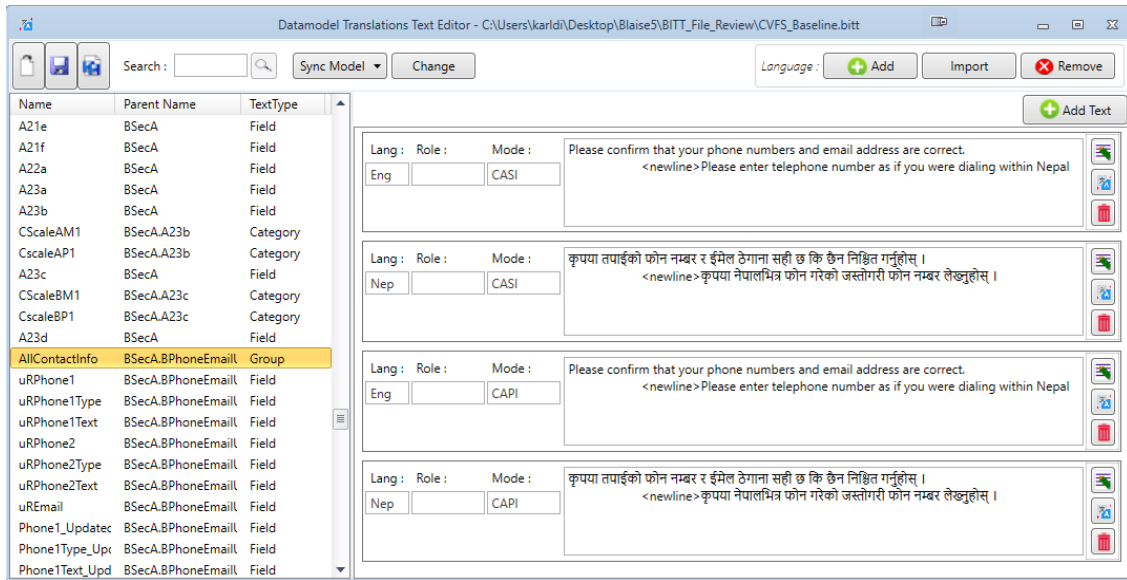
This BITT is a translation XML file containing the translatable text within a survey. It can be opened and edited using the Data Model Texts Editor (or the standalone "DataModelTextsEditor.exe" executable file). The file's XML structure is shown below in Figure 4.2.

Figure 4.2: Example BITT file

```
<?xml version="1.0" encoding="utf-8"?>
<TranslationsFile>
  <Languages>
    <Language Name="Eng">English</Language>
    <Language Name="Nep">Nepali</Language>
  </Languages>
  <Roles>
    <Role Name="Hint">A short explanatory text</Role>
    <Role Name="Help">A comprehensive instruction for the respondent</Role>
    <Role Name="Tooltip">Tooltip</Role>
    <Role Name="Answerinfo" />
    <Role Name="MethodHelp" />
    <Role Name="Template" />
    <Role Name="Templates" />
    <Role Name="Watermark" />
    <Role Name="PreLab" />
    <Role Name="PostLab" />
    <Role Name="EditMask" />
    <Role Name="MobileText" />
    <Role Name="Instructions" />
  </Roles>
  <Modes>
    <Mode Name="CAPI" />
    <Mode Name="CASI" />
  </Modes>
  <Items>
    <Translation Name="CVFS" Type="Datamodel">
      <Texts Lang="Eng">Chitwan Valley Family Study (V#: ^{Version})</Texts>
      <Texts Lang="Nep">चितवन उपत्यकाको पारिवारीक अध्ययन (संस्करण#: ^{Version})</Texts>
    </Translation>
    <Translation Name="AllMonths" Type="SpecialAnswer">
      <Texts Lang="Eng">All Months</Texts>
      <Texts Lang="Nep">सबै महिनाहरु</Texts>
    </Translation>
    <Translation Name="NoMonths" Type="SpecialAnswer">
      <Texts Lang="Eng">None</Texts>
      <Texts Lang="Nep">कुनैपनि होईन</Texts>
    </Translation>
    <Translation Name="SelectAll" Type="SpecialAnswer">
      <Texts Lang="Eng">Select All Test</Texts>
    </Translation>
    <Translation Name="CATIOfflineSkip" Type="SpecialAnswer">
      <Texts Lang="Eng">&lt;IWER&gt;Web Empty Response&lt;/IWER&gt;</Texts>
      <Texts Lang="Nep">&lt;IWER&gt;Web Empty Response&lt;/IWER&gt;</Texts>
    </Translation>
  </Items>
</TranslationsFile>
```

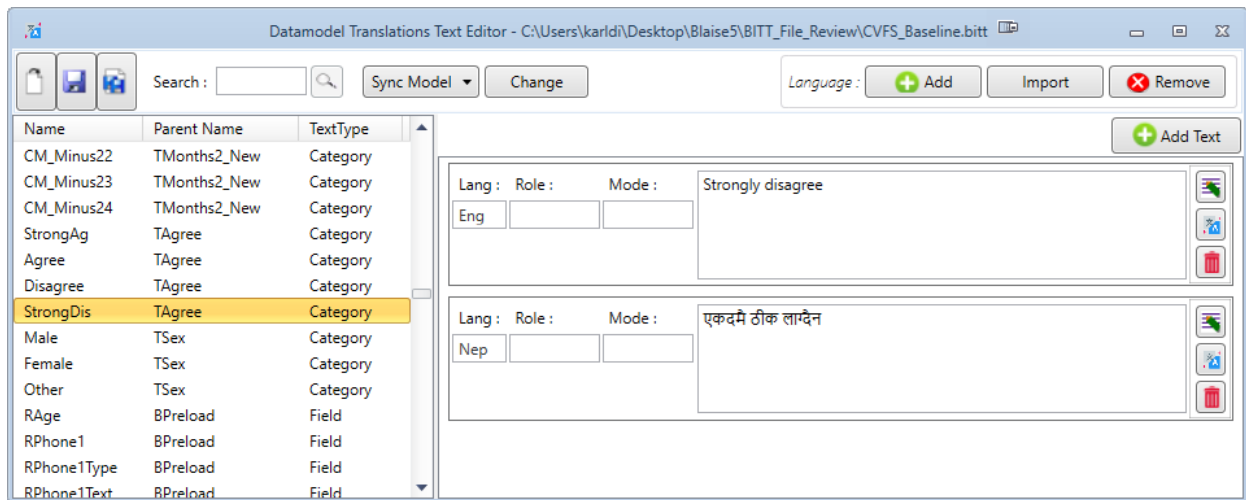
The following figures show the Data Model Text Editor's look and feel.

Figure 4.3: Blaise Data Model Text Editor - Example Question



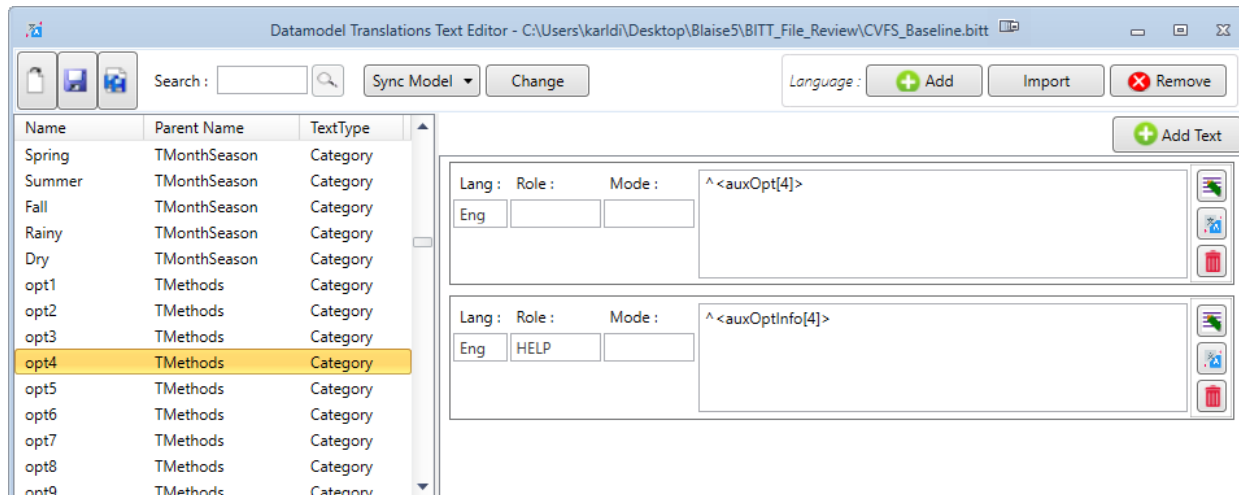
One drawback of the out-of-the-box solution in Blaise is that it is easy to lose context as the response options are broken into individual elements and not presented inline with the question text. Figure 4.4 below shows a response option for an agreement scale of Strongly Agree, Agree, Disagree, and Strongly Disagree. It is preferred to have all the text for a given question and response options presented together in one place.

Figure 4.4: Blaise Data Model Text Editor - Response Option Example



Additionally, fills are challenging to handle. Figure 4.5 below shows what response options look like when fills are used as the text of the response option.

Figure 4.5: Blaise Data Model Text Editor - Response Option as a Fill



The Blaise Data Model Texts Editor provides several useful features, summarized from the Blaise 5.15.2 online help documentation (2025):

- **Text Editing Capabilities:**
 - Users can edit or add texts for various Blaise elements, including: SpecialAnswers, Modes, Type, Const, Fields, Block, Group, and Checks.
- **Key Editor Features:**
 - Search: Quickly find specific texts or variables within the instrument.
 - Change: Globally rename or replace texts (e.g., translating from one language to another).
 - Validate: Ensure consistency of translations with both the data model and the underlying BITT XML file.
- **Language Management:**
 - Easily add new languages by specifying an ID and description.
 - Optionally copy texts from an existing language into a newly added one.
 - New languages are automatically recognized during model preparation without modifying the source code.
- **Potential Limitations of Blaise’s Out-of-the-Box Solution:**
 - Texts within the Rules section of the Blaise code (e.g., edit checks, fill texts) are not directly extracted. This limitation can be addressed using new features in Blaise 5:
 - **Const (Constant):** Defines fields with fixed values that cannot change over their lifespan.
 - **Checks Section:** Describes logical relationships between fields, auxfields, locals, or constants, or monitors specific fields for unusually high or low values, triggering errors if the specified conditions aren’t met.
 - However, these new features are less helpful for panel surveys that rely on existing programming standards established before their introduction.
 - **One-way Export Considerations:**

- Once texts are extracted, they remain stored in the BITT file. Newly added translations do not automatically feed into the original Blaise source code. While synchronization features exist, managing updates can sometimes become complex or cumbersome.
- Maintaining translated texts close to the source facilitates more straightforward management of panel surveys and surveys with evolving requirements over time.

The University of Michigan has utilized the BITT file to efficiently extract and organize text into a spreadsheet for translation purposes. However, due to the cumbersome nature of the BITT file, we have consistently resorted to manually copying the translations back into the Blaise source code. The Blaise source code has always served as the final repository for our translated text. One of the most significant challenges in translating surveys involves question fills. The complexity of these fills increases with the complexities presented by the target language. For instance, simple fills in English can become more intricate in languages like Spanish, where it is necessary to account for gender-specific word variations.

5. Reviewing our needs

Below, we review our basic requirements for the tool, the standards we envision using, and a quick review of some off-the-shelf tools that translation teams could use.

5.1 Requirements

Our specifications and the features needed are as follows:

- Follow the basic XLIFF principles of text extraction and the extract-translate-merge model (Savourel, 2001). Eventually, making the entire round trip:
 - Extract the translatable text and convert it to an XLIFF document.
 - Supply a valid XLIFF document for translation that can be loaded into a Computer-Assisted Translation (CAT) tool to serve as a workspace for the translation team.
 - Receive the updated XLIFF document containing the newly translated content and merge it into the Blaise source code.
- Extract as much text as possible (fill and edit check text) and present the text in the context where the text is used.
- Extract response options inline with the question text to offer the translator context for how the text needing translation is used.
- The tool should drastically reduce the burden of reincorporating the translated text into the Blaise source code.

5.2 Standards

The XLIFF Standard has several different versions. Version 1.0 was released in 2002, followed by versions 1.1 in 2003 and 1.2 in 2008. Version 2.0 was first released in 2014, succeeded by 2.1 in 2017, and the latest version, 2.2, will be released in 2025. XLIFF version 1.2 is the most widely supported and used version today. Apple, Adobe, and Angular use it. As a localization industry standard, it has broad support for features like Machine Translations and capturing notes about translations.

5.3 Off-The-Shelf Tools

Using an industry standard of XLIFF allows one to use off-the-shelf Computer Assisted Translation (CAT) tools, some of which are free to translate our content. Some of these tools are as follows:

- **OmegaT** (<https://omegat.org/>) is an open-source CAT tool that uses the Okapi Framework (<https://okapiframework.org/>) and a free translation memory tool.
- **MateCat** (<https://www.matecat.com/>) is a free and open-source online CAT tool.
- **Translate Toolkit** (<https://lokalise.com/solutions/xliff-translation>) a subscription-based product.

Some additional features that come into play when we begin thinking about using an industry standard tool are:

- **Translation Management Systems (TMS)** is a global management software application that aids in automating many parts of the localization process, allowing translators to work more efficiently. ("Translation management system," 2025)
- **Machine Translation (MT)** uses computational techniques to aid translation. These techniques could be used as a first pass before a translator reviews and finalizes texts ("Machine translation," 2025).
- **Translation Memory (TM)** is a database that holds “segments”, which can be in the form of sentences or other units of words that can aid a translator in doing their job. ("Translation memory," 2025).

Although these features are beyond the scope of this paper, we mention them as ways to enhance one's translation workflow when working with XLIFF.

6. Our solution

Our work builds on the work done by Ostergren, J. (2012), outlined in their paper "Tokenising Blaise Syntax via Regular Expressions". At conception, this tokenizing was used to duplicate language texts to ease the implementation of adding additional languages within the Health and Retirement Study (HRS). Since then, the tool has been adapted to include additional HRS-specific tasks that have saved the programming team countless hours of development time. After reviewing the tool, it was decided that the tokenization engine would assist our current efforts to extract the Blaise text needing translation. The tool was written specifically for the HRS Blaise instrument (which is one of the most complex instruments developed in Blaise), and its abilities to parse, tokenize, and manipulate source code had never been tested on other instruments at the University of Michigan.

After minor refinements to the regular expression (RegEx) strings, we quickly saw that the tokenization engine generally worked without issues. The first significant branch point was classifying the metadata, the backbone of the features and functions. The existing tool paints with a "broad brush" in a way suitable for use within their context. However, a generalized approach with a fine brush was necessary to make a tool that can work in many different contexts. For example, in Figure 6.1, we see the difference before and after, where the "COR" is defined in the Roles section, and "ENG" is described in the Languages section at the highest level of the data model in the new tokenization engine.

Figure 6.1: Alterations to the Tokenization Engine

Old Tokenization	New Tokenization
1 [Type - Codeframe]: TOOOLDTOWORK	1 [Type - Codeframe]: TOOOLDTOWORK
2 [Type - Codeframe]:	2 [Type - Codeframe]:
3 [Type - CodeNumber]: (3 [Type - CodeNumber]: (
4 [Type - CodeNumber]: 6	4 [Type - CodeNumber]: 6
5 [Type - Codeframe]:)	5 [Type - Codeframe]:)
6 [Type - Codeframe]:	6 [Type - Codeframe]:
7 * [Type - Codeframe]: COR	7 * [Type - Role]: COR
8 * [Type - Codeframe]: {CR}	8 * [Type - Comment]: {CR}
9 * [Type - Codeframe]: ENG	9 * [Type - Language]: ENG
10 * [Type - Codeframe]:	10 * [Type - Language]:
11 * [Type - Codeframe]: "<text key=vol>Too old to work"	11 * [Type - TypeText]: "<text key=vol>Too old to work"

The new classification keywords of "Mode," "Role," and "Language" are core Blaise syntax elements used in the source code of every Blaise data model; as part of the processing, these sections are parsed with the defined keywords pulled out (COR, ENG) and added to a programmatically accessible set per identifier if these keywords are encountered in the parsing of the code, we assign the corresponding identifier. This approach allows us to generalize to the Blaise syntax instead of hard-coding around a specific instrument configuration. Looking at the Blaise help file (2025) and the "Fields (section)," we see the following outline of syntax elements:

Figure 6.2: Blaise Fields Section

```

Syntax
FIELDS
Q1 [, Q2 [, ... ] ]
[ ( Tag ) ]
[ [ MODE ][, ... ]
  [ [ QUESTION ] [ LanguageId ] "Question text" [ ... ] ]
  [ { / | DESCRIPTION } [ LanguageId ] "Descriptive text" [ ... ] ]
  [ RoleName [ LanguageId ] "Role text" [ ... ] ]
  [ ... ]
[ ... ] ]
: FieldType [ "Constraint" ] [ , Attributes ]
[ ... ]

```

Once we determined the tokenization engine would meet our needs, we began working toward a Minimum Viable Product (MVP) for demonstration purposes to an internal team of stakeholders. While fragile and small in scope, this MVP could parse the demo survey, the Multilingual Flight Survey that ships with Blaise. Working towards successfully parsing question texts, collating them, and transforming them into a valid XLIFF 1.2 file (shown below in Figure 6.3). After transformation, the file is loaded into a freely available Computer-Assisted Translation (CAT) tool (shown in Figure 6.4) and worked on in the meeting in real-time, demonstrating a prototypical workflow that our translation staff can use. After the text is altered in the XLIFF-aware application, it is saved and returned to the Blaise programming team at the University of Michigan. The XLIFF file containing the newly translated target language text is then merged into the Blaise source code, completing the "round trip."

Figure 6.3: Example XLIFF File Created with the MVP

```
<?xml version="1.0" encoding="utf-8"?>
<xliiff version="1.2" xmlns="urn:oasis:names:tc:xliiff:document:1.2">
  <file original="Flight.blax" source-language="EN" target-language="ES" datatype="plaintext">
    <body>
      <trans-unit id="FirstName">
        <source>"&lt;inst&gt;What is your first name?"</source>
        <target>"Hello world"</target>
      </trans-unit>
      <trans-unit id="DateOfBirth">
        <source>"What is your date of birth?"</source>
        <target>"¿Cuál es su fecha de nacimiento?"</target>
      </trans-unit>
    </body>
  </file>
</xliiff>
```

Figure 6.4: Free tool <https://xliff.brightec.co.uk/>

Translations for Flight.blax

1 - "What is your first name?"

(ID: FirstName)

"Hello world"

2 - "What is your date of birth?"

(ID: DateOfBirth)

"¿Cuál es su fecha de nacimiento?"

We quickly iterated the MVP to support more complex surveys and added several features. Currently, the tool can parse our most intricate panel surveys, representing some of the most complicated instruments at the University of Michigan, while extracting question text and enumerated type texts. These enumerated type texts are displayed contextually alongside the extracted question texts, either beneath them or inline, as outlined in the requirements in section 5.1. This allows translators to modify these texts to ensure consistency with the question texts and the other questions and response options in the shared section.

Extracting different types of texts is beneficial, not only from a holistic question perspective but also because it enables individual languages to control the texts displayed on screen precisely. For instance, a question type of "TYesNo," where 1 = "Yes" and 5 = "No," may be appropriate in the English instrument; however, other languages may require additional context for the options to make sense in the target language. Our next step is to develop a standard naming convention for "duplicate types" and implement these types programmatically through the tool.

Additionally, our tool differs from the Blaise BITT file process in that it creates a single file for both source and target languages. For example, if an instrument has five languages, it generates five files containing the source language (typically English) alongside space for the five target languages. This is crucial because the person translating the Russian text might not have the same context or understanding as the person translating another language, highlighting the need for careful consideration in our tool's design.

At the time of this print, we have not yet addressed the question fills; however, this will be one of our next priorities. We acknowledge that this process will require several iterations to perfect. Our goal is to reach a stage where we no longer need to cut and paste the foreign-translated text into Blaise instruments manually. Nevertheless, we expect some manual work as we resolve ongoing issues.

7. Future considerations

We may need to update our programming guidelines to enhance our parsing efforts. For instance, we should revise the use of Blaise Constants within question fills and edit check text, as these may be easier to extract than other text found in the rules section of the instrument. Additionally, type names should be globally unique, and specific naming conventions must be limited to programmers and reserved for programmatic implementation. We will aim to replace programmatically defined auxiliary fields with texts defined as Constants whenever possible. Furthermore, we plan to upgrade our system to support XLIFF versions 2.1 or 2.2. Lastly, we want to review the translatable texts in the Blaise Resource Database (BLRD). While these translatable texts are beyond the scope of our current efforts, we intend to review them in the future.

8. Acknowledgments

The authors thank Jason Ostergren for jump-starting our work with the tokenization engine done for the HRS programming team.

9. Conclusion

As emphasized in the foundational work by Harkness et al. (2007) and the call for integrated translation tools by Dept, Ferrari, and Wäyrynen (2010), the survey research field has long needed a solution to support translation and localization, especially since the rise of computer-assisted interviewing. Ask anyone who has painstakingly copied and pasted multiple languages into a Blaise survey. Today, we're excited about our work and the multilingual projects on the horizon. This moment feels like the culmination of two decades of progress—a full-circle achievement that brings early vision to present reality. And it's just the beginning, with a promising future on the horizon.

10. References

Blaise 4 Help (Version 4.8..6.1960). (2017).

Blaise 5 Help (Version 5.15.2.3878). (2025).

Dept, S., Ferrari, A. and Wäyrynen, L. (2010). Developments in Translation Verification Procedures in Three Multilingual Assessments: A Plea for an Integrated Translation and Adaptation Monitoring Tool. In *Survey Methods in Multinational, Multiregional, and Multicultural Contexts* (eds J.A. Harkness, M. Braun, B. Edwards, T.P. Johnson, L. Lyberg, P.P. Mohler, B.-E. Pennell and T.W. Smith). <https://doi.org/10.1002/9780470609927.ch9>

Dinkelmann, K., Sparks, P., Ostergren, J., and Cheung, G. (2015). The Michigan Questionnaire Documentation System (MQDS) Version 5 Update. *The 16th International Blaise Users Conference (IBUC)*, Beijing, China.

Gager, J., Gregory, A., Nelson, C., Thomas, W. (2007). Data Documentation Initiative (DDI) Technical Specifications Part II: High-Level Documentation (Version 3.0) For Public Review Copyright DDI Alliance (Feb, 2007) <http://www.ddialliance.org>.

Harkness, J., Blom, A. (2006). 'ESS: Round 3 Translation Guidelines'. ESS Docs. <http://www.europeansocialsurvey.org/>

Harkness, J.A., Braun, M., Edwards, B., Johnson, T.P., Lyberg, L., Mohler, P.P., Pennell, B. E., & Smith, T.W. (2010). Comparative Survey Methodology. In *Survey Methods in Multinational, Multiregional, and Multicultural Contexts* (eds J.A. Harkness, M. Braun, B. Edwards, T.P. Johnson, L. Lyberg, P.P. Mohler, B.-E. Pennell and T.W. Smith). <https://doi.org/10.1002/9780470609927.ch1>

Harkness, J. A., Dinkelmann, K., Pennell, B. E., & Mohler, P. Ph. (2007). Final report and translation tool specifications (Deliverable No. 5), European Social Survey Infrastructure (ESSi) project: Improving social measurement in Europe, Mannheim: ZUMA.

Harkness, J. A., Dorer, B., & Mohler, P. Ph. (2010). Translation chapter. Guidelines for Best Practice in Cross-Cultural Surveys. Ann Arbor, MI: Survey Research Center, Institute for Social Research, University of Michigan. Revised 2016 by Mohler, P. Ph., Dorer, B., de Jong, J. & Hu, M. Retrieved March, 25, 2025, from <https://ccsg.isr.umich.edu/chapters/translation/>.

Harkness, J.A., Villar, A. & Edwards, B. (2010). Translation, Adaptation, and Design. In *Survey Methods in Multinational, Multiregional, and Multicultural Contexts* (eds J.A. Harkness, M. Braun, B. Edwards, T.P. Johnson, L. Lyberg, P.P. Mohler, B.-E. Pennell and T.W. Smith). <https://doi.org/10.1002/9780470609927.ch7>

Johnson, T. P., Pennell, B. E., Stoop, I. A. L., & Dorer, B. (2018). *Advances in comparative survey methods: Multinational, multiregional and multicultural contexts (3MC)*. Hoboken, NJ: John Wiley & Sons.

Language localisation. In Wikipedia, The Free Encyclopedia. Retrieved March 22, 2025. https://en.wikipedia.org/wiki/Language_localisation.

- Machine translation. In Wikipedia, The Free Encyclopedia. Retrieved March 22, 2025. https://en.wikipedia.org/wiki/Machine_translation.
- OmegaT. In Wikipedia, The Free Encyclopedia. Retrieved March 19, 2025. <http://en.wikipedia.org/wiki/OmegaT>.
- Ostergren, J. (2012). Editing Source Code Programmatically and Tokenising Blaise Syntax via Regular Expressions. *The 14th International Blaise Users Conference (IBUC)*, London, UK.
- Savourel, Y. (2006). Internationalization and Localization Markup Requirements, World Wide Web Consortium, Working Draft WD-itsreq-20060518, May 2006. <http://www.w3.org/TR/itsreq/>.
- Savourel, Y. (2001). XML Internationalization and Localization, Sams, Indianapolis, IN.
- Savourel, Y., Kosek, J. & Stoick, D. (eds.) (2008). Best Practices for XML Internationalization: W3C Working Group Note 13 February 2008. Retrieved March 19, 2025. <http://www.w3.org/TR/xml-i18n-bp>.
- Survey Research Center. (2016). Guidelines for Best Practice in Cross-Cultural Surveys. Ann Arbor, MI: Survey Research Center, Institute for Social Research, University of Michigan. Retrieved March 25, 2025, from <http://ccsg.isr.umich.edu/>.
- Smith, T.W. (2010). The Globalization of Survey Research. In Survey Methods in Multinational, Multiregional, and Multicultural Contexts (eds J.A. Harkness, M. Braun, B. Edwards, T.P. Johnson, L. Lyberg, P.P. Mohler, B.-E. Pennell and T.W. Smith). <https://doi.org/10.1002/9780470609927.ch25>
- Translation management system. In Wikipedia, The Free Encyclopedia. Retrieved March 22, 2025. https://en.wikipedia.org/wiki/Translation_management_system.
- Translation memory. In Wikipedia, The Free Encyclopedia. Retrieved March 22, 2025. https://en.wikipedia.org/wiki/Translation_memory.
- Willis, G.B., Kudela, M.S., Levin, K., Norberg, A., Stark, D.S., Forsyth, B.H., Brick, P.D., Berrigan, D., Thompson, F.E., Lawrence, D., & Hartman, A.M. (2010). Evaluation of a Multistep Survey Translation Process. In Survey Methods in Multinational, Multiregional, and Multicultural Contexts (eds J.A. Harkness, M. Braun, B. Edwards, T.P. Johnson, L. Lyberg, P.P. Mohler, B.-E. Pennell and T.W. Smith). <https://doi.org/10.1002/9780470609927.ch8>
- XLIFF Version 2.1. Edited by David Filip, Tom Comerford, Soroush Saadatfar, Felix Sasaki, and Yves Savourel. 13 February 2018. OASIS Standard. <http://docs.oasis-open.org/xliff/xliff-core/v2.1/os/xliff-core-v2.1-os.html>. Latest version: <http://docs.oasis-open.org/xliff/xliff-core/v2.1/xliff-core-v2.1.html>.