Digital infrastructure and its impacts on language work: A case study of FieldWorks Language Explorer (FLEx)

Authors

Abstract

Linguists need to collect, organize, analyze, and share data. Despite the variability of linguistic research (inputs and outcomes, etc.), there are some common stages, including morphological parsing, lexicon creation, and textual editing. A common software tool for this work is FieldWorks Language Explorer (FLEx). In this article, we examine some of the ways that FLEx's data structures conflict with contemporary practices in linguistic fieldwork. These assumptions harm the field, exclude key groups of users, and erect barriers to theoretically and practically relevant research. Where workarounds are feasible, they are both fragile and costly. We use this example to start a broader conversation about software infrastructure for digital linguistic fieldwork, and how tools that are designed to facilitate research may inhibit it.

1 Introduction

1.1 Preliminaries

This paper concerns the current state of digital infrastructure for language documentation and linguistic fieldwork. Using Fieldworks Language Explorer (FLEx) software (Summer Institute of Linguistics 2022) as an example, we explore ways in which software design and data structure decisions impact linguistic data workflows and erect barriers to research.

Linguists need ways to collect, organize, analyze, and share data – processes that may be called a project's "workflow", "pipeline", or "data life cycle" (cf. Rehm 2016; Gonzalez et al. 2017; Mattern 2022). Documentation and field methods handbooks dedicate much space to discussion of workflow (Bowern 2015; Chelliah & Reuse 2011; Meakins, Green & Turpin 2018), and documentation-oriented journals and conferences often include contributions describing new or adapted workflows for fieldwork data collection.

This emphasis on workflow arises because language documentation projects vary enormously in their inputs and outputs. Projects rely on different types of data – text, speech, or sign, individually or jointly (Di Carlo, Ojong Diba & Good 2021; Hochgesang 2022; Cox 2022; Lukaniec 2022). They assemble datasets of very different sizes: for example, audiovisual documentation corpora range from fewer than 10 to more than 1,000 hours of recording time (Palfreyman 2022; Skilton 2021). Many collections are eclectic, including different modalities and participation frameworks (Caballero 2017; Gawne 2018). These varied inputs call for different workflows. Furthermore, because analysis often leads to data modification, there is no single unidirectional pathway that data takes through the lifecycle of documentation work. Documentation projects also produce various outputs, ranging from reference grammars to classroom materials to storybooks. This diversity of goals leads to even more variation in workflow.

Underneath this variation, what contemporary language documentation workflows have in common is a reliance on digital infrastructure. Many projects use entirely digital pipelines and produce primarily digital outputs, such as video games (Grassroots Indigenous Multimedia 2023), online courses (Chew, Lokosh & Morgan 2022), or animations (Silva 2016). The digital infrastructure for language documentation includes tools such as Praat (Boersma & Weenink 2022), ELAN (Wittenburg et al. 2006), and FieldWorks Language Explorer or FLEx; less-common tools include LaMeta (Hatton et al. 2021) and AntConc (Anthony 2022). Many of these tools – for example, ELAN and Praat – align with the recommendations given by Bird and Simons (2003): they write data in open-source, transparent, and human-readable formats, such as XML. As a result, their outputs have proven relatively persistent and portable to other formats. Some fieldwork apps, such as Praat, have remained more or less unchanged since the 1990s exactly because they adhere to these principles.

Minimizing changes to digital infrastructure works in favor of portability, yet the range of computational work that is possible with language data has expanded enormously in the last 30 years - especially in the last 10. Increases in processor power, decreases in the cost of storage, and the advent of born-digital video recording have all transformed the possibilities for data collection and annotation. Likewise, the global availability of smartphones has created an almost entirely new set of pipelines for remote or "distributed" fieldwork (Williams et al. 2021; Grzech & Tisalema Shaca 2022).

Alongside these changes in methods, recent work in language documentation also shows a greater awareness of how documentation experts need to respond to the extractive past and present of the discipline. While community-oriented and community-led linguistics has been discussed at least since Wilkins (1992), academic work increasingly centers Indigenous approaches to language and healing, language advocacy, and language renewal (McCarty et al.

2018; Whalen et al. 2022; Leonard 2017; Woods 2023). These approaches emphasize embedding language in cultural, social, environmental, and artistic practices. They also foreground collaborative practices – where language connects individuals and families with elders and the voices of ancestors – and intergenerational transmission. Scholars in this tradition simultaneously place a renewed emphasis on reclamation of archival materials, critiquing the aims of salvage documentation and extractive linguistics (Davis 2017).

Our focus here is on one particular aspect of the fieldwork dataflow, involving the transcription and annotation of text; morphological parsing or glossing; and lexicon building. To support this part of the dataflow, many researchers in language documentation rely on Fieldworks Language Explorer (FLEx), a linguistic analysis and database management software package. FLEx is widely used because it brings together a powerful morphological parser and features that allow the user to compile dictionaries. However, FLEx also makes many – more or less implicit – assumptions about the user and their computer; the language data; the (morphological) structure of the study language; and the types of output required after analysis.

We argue that these assumptions exclude key groups of users and erect barriers to theoretically and practically relevant research. While this article is focused on FLEx, we emphasize that our argument is more general. As a field, language documentation relies on rigid and outdated digital infrastructure. All tools embed assumptions in their data structures that inhibit some types of research and facilitate others. That is unavoidable. But in the case of language documentation software, these implicit assumptions force users to develop time- and skill-intensive workarounds which would be unnecessary with a more flexible and more linguistically informed software infrastructure. We highlight these issues for FLEx specifically because of its centrality in morphological analysis workflows and its use in multiple types of research, but similar points can be made about other software.

We begin in §§1.1-1.3 with a discussion of FLEx software and its aims, before moving in §2 to general issues that shape projects across many linguistic subfields. Section §3 describes specific use cases.

1.2 What is FLEx?

Fieldworks Language Explorer (FLEx) is a linguistic software suite for data organization and analysis. It is produced by the Summer Institute of Linguistics, a Christian faith-based organization focused on language and Bible translation.[1] FLEx is widely used for analysis of Indigenous language materials. The software is based on Toolbox, an earlier SIL-produced package with similar functions, and replaced Toolbox in 2006-2007 (Butler & van Volkinburg

[1] From SIL International's mission statement (www.sil.org/about): "Inspired by God's love, we advocate, build capacity, and work with local communities to apply language expertise that advances meaningful development, education, and engagement with Scripture."

2007). FLEx is frequently[2] updated, has always been free to use, and as of v. 9.0 (released September, 2018) is open source.

FLEx combines three main modules: a lexical database, a database of texts, and a morphological parser. Users can gloss texts either by constructing a lexicon and then running the parser, or by building the lexicon incrementally as they analyze texts (i.e., by training the parser). The lexicon provides structured fields for creating a bilingual dictionary, including a one-way look-up function for finding examples in texts to add to the lexicon. The parsing module allows the use of two algorithms: a default string-based parser (XAmple) and a parser that incorporates phonological rules (HermitCrab).[3]

There are no viable competitors to FLEx that combine all of these functions. Though other dictionary creation and text analysis packages exist, they are generally produced by individual scholars or small teams (Dunham, Cook & Horner 2014; Sande & Ewert 2023), meaning that they (a) do not offer technical support and (b) may cease to be maintained when their creators shift their professional focus. The closest competitor is ELAN (Wittenburg et al. 2006), a transcription program that provides support for corpus search and has basic interlinearization capabilities. In theory, because ELAN is institutionally supported by the Max Planck Society, it may be a viable competitor. But as of this writing, ELAN's Interlinearization Mode is in beta and lacks key features, such as allowing the user to add new lexical items from the transcript text. ELAN also cannot accommodate text-only data without linked media. These issues mean that ELAN cannot yet compete with FLEx as an interlinearization tool. A competitor in the dictionary creation market is TshwaneLex, a widely used commercial bilingual dictionary program with deep discounts for academic and community use. TshwaneLex has corpus linkage functions but lacks a parser, making it unsuitable for sole use in language documentation. Other stand-alone parsers exist, but they are typically trained for particular languages, require knowledge of Python, or do not integrate into a language documentation workflow.

1.3 FLEx's user aims

Toolbox and FLEx were originally developed for Bible translation (https://web.archive.org/web/20110912212755/www.sil.org/computing/shoebox/TransQA.html), to aid the translator collecting texts and developing a lexicon from them for the translation (Dobrin & Good 2009). Today, however, SIL describes FLEx as "software tools that help you manage linguistic and cultural data."[4] More specifically, the organization states that FLEx is designed for "dictionary development, interlinearization of texts, morphological analysis, and

[2] https://software.sil.org/fieldworks/release-notes/

[3] Parsing in FLEx is described in Black et al. (2007).

[4] Quackenbush (2007), for example, suggests that the faith-based elements of SIL's program combine with other activities that together promote community wellbeing. The training site LingTranSoft (https://lingtran.net/LingTranSoft-Wiki) also closely integrates FLEx and dictionary training with SIL's Biblical software, such as ParaText.

[preparation of] other publications" (https://software.sil.org/fieldworks/). SIL's software repository offers a more specific list of tasks that FLEx can accomplish, stating that researchers can use the app to "identify the most frequent words in a natural text corpus," "run a concordance search on a natural text corpus," and "publish linguistic data in a structured way" (https://lingtransoft.info/apps/flex-fieldworks-language-explorer).

As this description suggests, the major functions of FLEx are to collect language samples – usually called "texts" in documentation – and parse them to produce interlinear glossing. Based on the parsed materials, users can compile information about lexicon and morphology and produce bilingual dictionaries in the language of study and the contact language. Thus, the design structure of FLEx is deeply intertwined both with (Bible) translation as a means and goal of data collection, and with the "Boasian trilogy" of grammar, dictionary, and monologic texts as the primary output of language documentation (cf. Musgrave & Thieberger 2021).

Additionally, FLEx and Toolbox were developed during the discipline's transition from analog to digital fieldwork, and this is evident in their design structure. Analog multimedia data is inherently unlinked except by manual cross-references.[5] In contrast, most digital fieldwork pipelines rely on multidirectional dataflows, where transcriptions can be continuously updated based on analyses and vice versa. This "back and forth" is a major advantage of digital language documentation but is mostly absent in FLEx: for instance, changing the form of a morpheme during parsing in FLEx does not change the morpheme's representation in the baseline.

2 General Issues

FLEx makes a variety of assumptions about the user and their computer; the writing systems of input text; and multilingual content in text data. Because these issues potentially affect all of the types of projects described in §3, we review them together. We begin with a review of the technical requirements for running FLEx, then moving to project setup issues (writing systems and multilingual content), before closing this section with a discussion of collaboration and integration of language documentation and reclamation.

2.1 Technical Requirements and Accessibility

The technical requirements of FLEx create a variety of obstacles for users. Because FLEx is available only for Windows and Linux, Mac-based researchers cannot run FLEx on their own computers without a virtual machine or remote desktop. Furthermore, because FLEx requires a disproportionately large amount of RAM, it performs poorly on virtual machines and remote desktops. Similar performance issues appear when using FLEx on a Windows machine with low RAM, such as a netbook. Additionally, FLEx uses significant processing power and energy and

[5] For example, once a cassette tape was transcribed, data analysis typically happened only with the transcribed materials, with minimal reference back to the audio information.

can rapidly exhaust laptop batteries. The program also leaves processes running – draining the power source and slowing performance – even after the user quits FLEx.

Some researchers work around the OS requirements by purchasing dedicated computers used only for database work. While this can reduce FLEx-related performance issues on the user's main computer, it is too expensive for many teams. Even for those who can afford a FLEx-specific computer, transporting and maintaining two computers during fieldwork can be impractical.

Besides these technical issues, FLEx also falls short in visual accessibility. The app is not compatible with screen readers and does not allow users to adjust the text size of the menu items, the text color, or background colors. This is especially problematic for accessibility because the default text size is relatively small and the background colors – light blue, yellow, and white – are bright and low-contrast.

Because of these issues, FLEx functions best for a user working on a single Windows computer that has plenty of RAM, a good CPU, and either a good battery or a connection to stable outlet power. In contrast, real language documentarians often work on non-Windows systems, use older or lower-power computers, run their computers on solar or battery power, and do not have the funds to buy new computers only for FLEx. The program is also designed for a user who has no need for visual accessibility features, but in reality, language documentation researchers often do require them, either because the researchers are visually impaired, or because they work with people, such as elders, who are.

2.2 Orthography and Fonts

FLEx allows users to choose between many standard orthographies or to create custom ones. Many language documentation researchers take the second option. While the option to create custom writing systems is helpful, the writing system setup function of FLEx lacks data validation features and is structured around a rigid division between word-forming characters and punctuation. As a result, even small oversights in writing system setup can produce errors in analysis at best and make text unanalyzable at worst.

When configuring a custom writing system in FLEx, users must designate every character in the system as either a word-forming grapheme or a word-delimiting punctuation symbol, as shown in Figure 1. These sets of characters must be mutually exclusive, which creates problems when the orthography includes grapheme characters that resemble, or can also be used as, punctuation. For example, suppose the user is setting up an orthography for a language which represents glottal stops with an apostrophe. In the FLEx writing system, the character representing the glottal stop must differ from the punctuation character used as an apostrophe or single quote. To assign these roles, the user will need to add each character manually to either the Word Forming category or the Punctuation, Symbols, & Spaces category. These assignments can be adjusted later, but inconsistencies are difficult to reconcile once text analysis has begun.

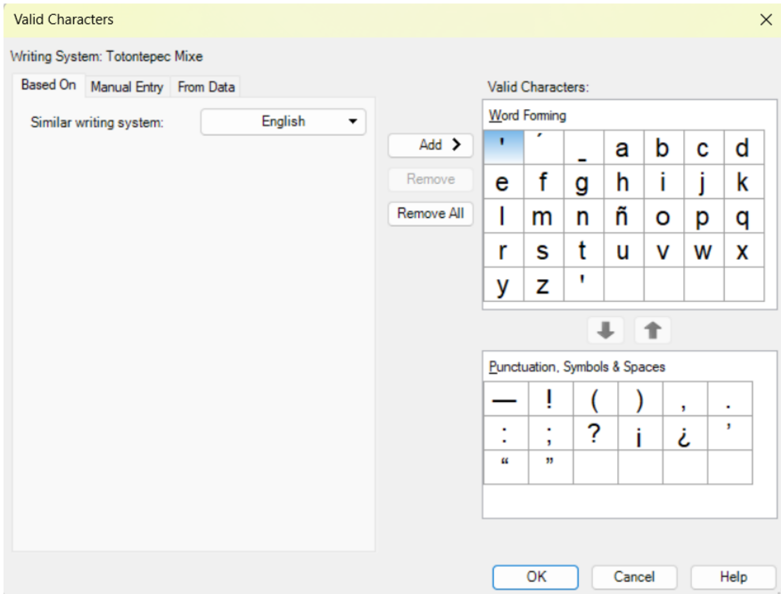

Figure 1. The writing system configuration pane in FLEx

After the glottal stop character is assigned as a word-forming character, if a glottal stop is written in a text with an apostrophe, FLEx will read it as punctuation and insert a word break wherever the character appears, as shown in Figure 2. This issue will make the text unanalyzable until the glottal stop character is replaced by a character that FLEx reads as a grapheme, as shown in Figure 3. Conversely, if a character that is intended to be punctuation is not explicitly designated as punctuation in the writing system, words adjacent to the character will be joined. The character will be treated as an element of the word and, by default, included in the morphological analysis, as shown for the <?> character in Figure 4.

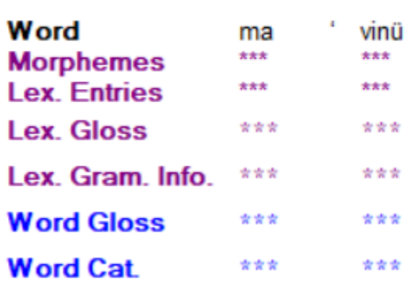

Figure 2. Graphemes read by FLEx as punctuation break up words

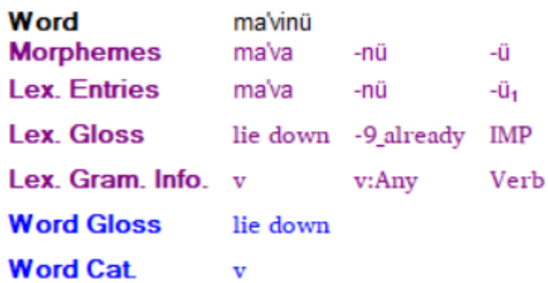

Figure 3. An apostrophe-like glottal stop grapheme properly read by FLEx as a word-forming character

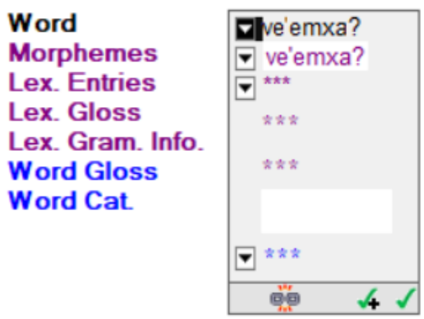

Figure 4. Punctuation characters read by FLEx as graphemes are incorporated into morphological analysis

Additional problems may arise if there are orthographic inconsistencies across a corpus. For example, if multiple keyboards are used to transcribe texts with the glottal stop character, the character may be represented with several different variants, such as the straight saltillo <ꞌ>, its uppercase value <Ꞌ>, or any of the three curled apostrophes <’ ‘ ‛>. Furthermore, FLEx lacks orthography validation features, meaning that users are not alerted when a character appears that is not found in the current writing system. This makes orthographic errors easy to miss. Additionally, while FLEx allows users to designate variants of morphemes and words, it does not allow them to designate variants of characters. Adding a character-variant feature would solve many of the orthography issues described in this section.

2.3 Multilingual Content

FLEx texts and databases can be multilingual, but the program’s data structure forces each line of a text (lines, termed “phrases” on the FLEx backend, often correspond to utterances) to be treated as monolingual. This means that, in turns containing intrasentential code-switches out of the study language, researchers must either leave the code-switched content unanalyzed or treat it as words of the study language. Either of these options could be inappropriate for building an analysis using the database. If a text contains code-switches <u>between</u> lines, those can be analyzed. But since FLEx does not come with any lexica, the user will need to manually add lexical entries for every item in the code-switched lines – even if the code-switched content is in English, Spanish, or another high-resource language.

If researchers do want to analyze intrasentential code-switches, a popular workaround is to enter all code-switched words as allomorphs of a single headword in the study language lexicon, called something like "EnglishWord" or "SpanishWord." Alternatively, if the user plans to study syntactic characteristics of code-switching, they can create unique "EnglishNoun" and "EnglishVerb" (or parallel) headwords. These workarounds allow users to search for code-switches, but they are labor-intensive: users must individually add each new code-switched word encountered in the transcripts as an "allomorph" of the relevant headword.

Another solution is to translate code-switched items back into the other language. This is a popular option for heavily edited texts for endangered language documentation materials, but the translation solution has drawbacks as well. It does not represent what language users actually say, and it divorces the text transcript from the underlying recordings, which causes problems if the text is later used in forced alignment or other work that requires an exact match between the audio/video and transcript.

2.4 Collaboration and Integration

FLEx is difficult to use in a collaborative research setting. Unlike much professional software used today, the app does not allow multiple users to edit a database simultaneously. While users can share databases with one another, the version control process is opaque, and lexical entries do not include version histories. Furthermore, remote collaboration requires either exchanging database backups or accessing the database via an account on LanguageDepot and/or LanguageForge, which must be manually created by SIL. As a result, in practice, only one researcher (or one machine) is able to work in the database at a time, unless they are working on non-overlapping areas. These constraints often mean that research teams spend a great deal of time reconciling edits, waiting to work until others are finished, or coordinating handoffs of FLEx-specific hardware. The lack of explicit versioning within FLEx and its arbitrary reversion of dictionary records makes reconciling joint edits time-consuming and error-prone. For example, if two users edit the same dictionary entry, at sync the conflict is resolved by keeping only one set of changes. The program generates a warning, but the warning does not say which changes have been kept, leaving the user to deduce what the additional changes might have been.[6]

As important, FLEx assumes that language can be studied independently of its social context. For the most part, this is not a controversial view for linguists – after all, the field of linguistics is built on the treatment of language as an independent, modular system. But language records do have extensive connections outside of language. They record information about material culture, philosophy, and history, and they encode cultural knowledge. They are records of interactions,

[6] There is an option to compare versions on the LanguageDepot site, but no option for a user to restore from intermediate points.

and can also be important mementos of individuals. It is therefore important that the context for language records can be included in any language documentation project. This can happen in several different ways: through rich metadata (Bird & Simons 2003; Sullivant 2020, among others), for example, and through making sure that language materials can be easily repurposed from FLEx into other formats, as needed. As the Indigenous Protocol and Artificial Intelligence Working Group (2020) put it, "Indigenous knowledge is often relational knowledge," so Indigenous projects that center language need to also be able to retain the many relations that language creates.

While FLEx is mostly used in language documentation and translation projects, it is not used only in academic settings. As discussed in §1.2, FLEx was developed primarily for translation and dictionary compilation. Furthermore, it was designed for language programs with a linguist working on language/linguistics, rather than the collaborative approach that many reclamation programs stress (see e.g. Hermes 2012). Nevertheless, such programs often make use of materials collected by community outsiders (see e.g. Warner, Luna & Butler 2007; Baldwin, Costa & Troy 2016; Lukaniec 2022) and need some way of organizing such materials in order to analyze and use them. The remainder of this section focuses on these issues from a language reclamation perspective.[7]

The relationship between language and wellbeing is central to many Indigenous language reclamation programs. That is, the goal of a language program is not simply to increase fluency in or prominence for the language, but to situate the language in a much larger network of health, culture, and Indigenous praxis (e.g. McCarty et al. 2018; Leonard 2017; Chiblow & Meighan 2022; Whalen et al. 2022). Likewise, while Buckskin et al. (2021) discuss how to give prominence to the Kaurna language within the City of Adelaide, their focus overall is on the process of increasing language knowledge as a step towards healing for members of the Kaurna community.

In Indigenous language reclamation programs, much importance is placed on methodology and collaboration while working on language data. The processes of bringing the language home, connecting with ancestral voices, and rehousing and resituating the knowledge within the language materials are deeply emotional and challenging – though often also ultimately restorative. Such points emerge repeatedly from contributions at the International Conference on Language Documentation and Conservation (ICLDC, http://ling.lll.hawaii.edu/sites/icldc/). One major theme in ICLDC contributions is that working on linguistic analysis fosters collaboration and belonging. That is, language reclamation is not simply about adding words to a database. Software that makes collaboration difficult also makes reclamation difficult.

[7] Our positionality as community outsider language researchers means we have limited experience in discussing the use of FLEx in language reclamation programs, and we draw on our experience in community collaborations, as well as from the literature on reclamation methods by Indigenous researchers, in discussing this topic.

Whether they work from legacy materials and databases or from contemporary speakers' knowledge, reclamation programs also typically place great emphasis on creating materials that can be circulated in and used by the Indigenous community. Because such materials are "for the community," it is important to be able to move materials into and out of the database in a variety of formats. As we discuss in §3.2 and §3.3.2 below, however, this import and export process is difficult to achieve in FLEx without extensive workarounds or reformatting.[8]

Indigenous sovereignty includes Indigenous intellectual property and language rights (see Indigenous Protocol and Artificial Intelligence Working Group 2020, among many others). Programs that require users to upload their data to a server that is not in their control, such as LanguageDepot or LanguageForge, run counter to these principles. As the members of the Indigenous Protocol and AI Working Group (2020: 22) write, "Indigenous communities must control how their data is solicited, collected, analysed and operationalized. They decide when to protect it and when to share it, where the cultural and intellectual property rights reside and to whom those rights adhere, and how these rights are governed. All AI systems should be designed to respect and support data sovereignty." While FLEx is not an AI system, these recommendations still matter: the issue of data sovereignty is not specific to AI.

In summary, the integration of language in the community is partly about the ways that cultural practices are documented through the language. But it is also a means for community members to reconnect with elders and ancestors about language. Such connections (whether interdisciplinary or inter-community) are also important for language documentation more generally, as repeatedly stressed in documentary linguistics handbooks (Chelliah & Reuse 2011; Meakins, Green & Turpin 2018). FLEx currently supports these connections only to a very limited extent.

3 Data Structures and Their Implications: Four Examples

The sections that follow describe examples of how the data structures defined by FLEx interact with language documentation and analysis workflows. These cases were chosen to illustrate the breadth of issues and the importance of this topic for language documentation.

3.1 Language with Nonconcatenative Morphology

In this first case study, we consider the use of FLEx for the analysis of a language with nonconcatenative morphology. Specifically, we explore the challenges that arise when using FLEx to track alternations in surface forms as a step towards analyzing their conditioning.

[8] Indeed, several software programs, including Miromaa (Miromaa Aboriginal Language & Technology Centre 2023) and Acorns (Harvey 2022), have been developed for reclamation work. These place much greater emphasis on the topics described here. However, neither has the parsing capabilities of FLEx.

FLEx makes many assumptions about language structure, particularly about the relationship between morphology and phonology. For research on languages with nonconcatenative morphology, these assumptions require labor-intensive workarounds and raise unnecessary barriers to analysis. Our goal in this section is to describe the impact of FLEx's assumptions about grammar, rather than its assumptions about the user or data structure, which are described in later use cases. Therefore, in this case we assume that the FLEx user is working from a set of monologic narratives, transcribed in a standardized orthography.

3.1.1 Advantages of FLEx

FLEx is optimized for monologic texts that have been compiled in paragraph format, such as in a word processor. Texts in this format can easily be copied and pasted into the baseline pane of the FLEx Texts & Words module. For texts in paragraph format, there are no further steps in the import process.

3.1.2 Disadvantages of FLEx

By default, FLEx uses a string-based parser called XAmple, which draws on the lexicon and surface forms of the analyses that the user has approved to recognize patterns and suggest future analyses (Black, Simons & Zook 2007). This parser functions somewhat like a field notebook for language description: with XAmple, FLEx can be used to store information as the researcher learns about the language, which, in turn, trains the parser to suggest previously-approved analyses for unanalyzed items, as long as these unanalyzed items are orthographically identical to ones that have already been approved.

Alternatively, FLEx offers a phonological rule-based parser called HermitCrab.NET (Maxwell 2003), which functions optimally when rules governing morphology are detailed and explicit. Unlike XAmple, this parser is generative and therefore presupposes fairly complete linguistic analysis to accurately predict exclusively well-formed analyses in novel parsed forms. As a result, there is little room for pure linguistic description as a means to arrive at the proper conclusions.

The choice between these two parsers will present obstacles to a researcher describing a language with nonconcatenative morphology. While HermitCrab is better-suited to systems of nonconcatenative morphology, it will not be a feasible option when the factors that condition an alternation are in question. On the other hand, while XAmple allows for rule-free description, it works best with agglutinative morphology that has minimal allomorphy.
Consider a case in which a researcher is describing a language that exhibits stem alternation and is interested in investigating whether the stem alternation is phonologically- or morphologically-conditioned. In this language, stems consist of CVC syllables; and vowel nucleus shapes and vowel qualities both alternate, such that the same stem can appear with either of two vowel

nucleus shapes and either of two vowel qualities. The alternations are crossed, meaning that stems have at least four distinct forms, as illustrated in Table 1.

Table 1. Stem alternation patterns

Frame: p_t	Nucleus Shape A	Nucleus Shape B
Vowel Quality A	peet	pe'et
Vowel Quality B	piit	pi'it

There are at least two options for tracking these alternations. First, to reflect all possible permutations of nucleus shapes and vowel qualities a given stem can have, the researcher may choose to create separate lexical entries, allomorphs, or variants for each iteration of that stem. In this language, each verb would then have four associated lexical entries or allomorphs. Alternatively, each of the possible nucleus shapes and vowel qualities may be assigned their own lexical entry (Table 2) and be marked on verbs as they appear throughout the corpus, using a placeholder concatenative morphological category already available in FLEx, such as an enclitic.

Table 2. Placeholder morphemes for tracking stem alternation in FLEx

Vowel nucleus shape or quality	Placeholder morpheme
/e/	=stem.e
/i/	=stem.i
V	=v.short
V:	=v.long

This second workaround will enable the researcher to keep track of all of the relevant categories in the corpus without requiring a complete analysis of their distribution. Nevertheless, this solution is inefficient. For every token of a lemma where the surface form does not match what is determined to be the headword in the lexicon, the researcher must manually change the surface form to the headword for the program to register the correct lexical item. Furthermore, this workaround obscures the morphophonological processes at work by relying on the framework of concatenation to classify nonconcatenative morphology. This may be particularly problematic if the language of study is also synthetic, as the researcher will need to ensure that concatenative morphology is distinguishable from these placeholder morphemes for further analysis.

On the other hand, if we suppose that the researcher is confident enough about the distribution of the stem alternations to formulate rules that predict novel analyses using HermitCrab, the process of setting up the appropriate rules to capture the morphophonology of the language may be just as time-consuming if not more so than the ad-hoc enclitic solution posed above. Depending on the morphological system in question, it may be necessary for the user to fully describe any combination of the phonemes, natural classes, phonological features, conditioning environments, verbal template, and inflection system, as well as all exceptions to these rules, in order for the parser to work.

3.1.3 Implications

While some of the attributes of FLEx that have been discussed in this section encourage certain elements of linguistic research, they discourage many others. These are outlined in Table 3.

Table 3. Implications of using FLEx for the analysis of nonconcatenative morphology

Attribute	Encourages	Discourages
Baseline structured in paragraphs, as in a text file	Copying and pasting directly from a text file	Import of text in any other format (cf. §3.3.2)
XAmple, FLEx's default string-based parser	Describing linguistic phenomena without a comprehensive analysis of the language Analyzing agglutinative morphology with little allomorphy	Analyzing nonconcatenative morphology
HermitCrab.NET, an alternative phonological rule-based parser	Implementing comprehensive, rule-based generalizations about conditioning of allomorphy Optimized for analyzing simple allomorphy with few conditioning factors or exceptions	Describing data in order to arrive at new generalizations about conditioning or output

3.2 Team Dictionary & Text Preparation

In this use case, we examine the advantages and disadvantages of FLEx for preparing dictionaries and text collections, two common types of language documentation publication. A common use of FLEx is where a team of researchers have created a lexicon and collection of texts, then aim to publish a dictionary and text collection that they can share with other members of the speaker and research community. For work with endangered language communities, materials should be bilingual and usable for learners of the language of study.

3.2.1 Advantages of FLEx

FLEx has many advantages for this use case. The app can house both lexical and textual materials, meaning that the dictionary can be built incrementally from the analyzed texts. The lexicon module also allows the user to link lexical entries to examples from the text collection, ensuring that example sentences are culturally relevant and representative of the language as used in the texts.

Within the lexicon module, users can add custom fields in addition to the fields provided by FLEx, and they can define custom categories in many built-in fields (e.g., custom parts of speech). This allows users to adapt the lexicon structure to the grammatical categories relevant to the language and to include cultural and encyclopedic information in the dictionary. Users can also batch edit and search lexical entries.

3.2.2 Disadvantages of FLEx

Dictionary Production

Though the FLEx lexicon module is customizable, its dictionary export process is rigid. Once the user has exported a dictionary from the lexicon, they cannot reorganize entries for the print output. They also cannot add supplemental material to the dictionary, except by manually inserting pages into the PDF. Users also have little control over the design or typesetting of the dictionary export: for example, they cannot add illustrations to specific lines of a dictionary entry. FLEx does allow customization of character and paragraph styles, but in our experience, they interact in unpredictable ways.

Text Production

FLEx’s export of interlinearized texts is similarly rigid. When the user exports an interlinearized text from the FLEx texts module, they cannot alter any line breaks. Nor can they edit the baseline to introduce new punctuation, because adding punctuation forces a line break, as illustrated in Figure 5. FLEx also does not support copying and pasting the baseline text. This removes the interlinearization, making it difficult to insert individual sentences into secondary documents, such as when quoting examples in a reference grammar.

Figure 5. A line in FLEx that does not have punctuation (left) is split into two when punctuation is added (right)

Some of these issues can be resolved using technical workarounds. Beier and Michael (2022), for example, describe a workflow that allows them to generate a customized dictionary, with front matter, from a FLEx database. Their workflow, however, relies on custom Python and LaTeX scripts in addition to FLEx. Few language documentation teams have the programming expertise necessary to write these add-ons. Moreover, even for teams with a programmer, the FLEx backend uses an extremely idiosyncratic XML structure (cf. the discussion of export formats in §3.3.2), creating a significant learning curve to writing scripts that interface with the program.

3.2.3 Implications

The comprehensive, customizable lexicography tools offered by FLEx encourage the user to invest a great deal of time in collecting and editing the lexicon. Though the text module is less well-designed - as discussed in our next case study - it also allows users to invest a great deal of effort in collecting and analyzing texts. But at the point of publication, the difficulty in editing the output of the dictionary and text exports discourages users from investing any time in the preparation of the dictionary or text collection as a product. This has been a problem since the first release of FLEx (cf. Butler & Van Volkinburg 2007).

In sum, the dictionary export process leads to a mismatch between the content of FLEx-based publications, which may be excellent and highly adapted to the language and culture, and their appearance, which will be poorly typeset, awkwardly laid out, and uncustomized. This unfinished appearance makes publications less likely to be used. It also sends a negative message about the importance of the publication - and the effort of the research team - to language community members and other end-users. Table 4 summarizes these implications of FLEx for dictionary and text preparation.

Table 4. Implications of using FLEx for collaborative preparation of dictionaries and texts

Attribute	Encourages	Discourages
Putatively comprehensive lexicography tool	Using FLEx as the central locus of lexicographical work	Branching ideas for other kinds of lexicographical work
Limited but straightforward export options for texts and dictionaries	Shipping off texts and dictionaries immediately after their analysis/curation	Detailed typesetting configurations (all options are formats that are difficult to modify out of the box) Respect for/Credence to the care put in to curate these resources

3.3 Transcript Analysis

In this third use case, we explore the use of FLEx for research on child language. Since many child language studies are based on recordings of child-caregiver interaction, most of our points also apply to research on conversation between adults. Child-language research data brings up many points that make digital data infrastructure concerns clear, including interoperability, customizability of fields, and customizability of data structures. That is, this use case crystallizes a number of issues that also appear in other subfields.

Consider the situation where a team of child language researchers works with recordings of child-caregiver interactions, made at consistent intervals over a year. ELAN (Wittenburg et al. 2006) and CLAN (MacWhinney 2000) are currently the standard software in this field for annotating such collections.

Recordings of child-caregiver interaction definitionally include at least two participants - the target child and their caregiver - and many also include other participants, such as siblings or other caregivers and adults. When annotating interactions, researchers attribute turns in ELAN or CLAN to speakers and label them for addressee type, as shown in Figure 6 (e.g., the “T” code in the “xds” tier in Figure 6 denotes a turn addressed to the target child). The speaker information allows users to distinguish the target child’s speech from other participants’, while the addressee information allows them to distinguish speech directed to children vs. to adults. Additionally, child language data always includes some child utterances that are ungrammatical in the adult grammar. This also impacts the ELAN/CLAN annotation structure: ungrammatical turns are labeled as ungrammatical and annotated with the equivalent adult form, as shown in the 'YDN_Target-mto-txt' tier in Figure 6.[9]

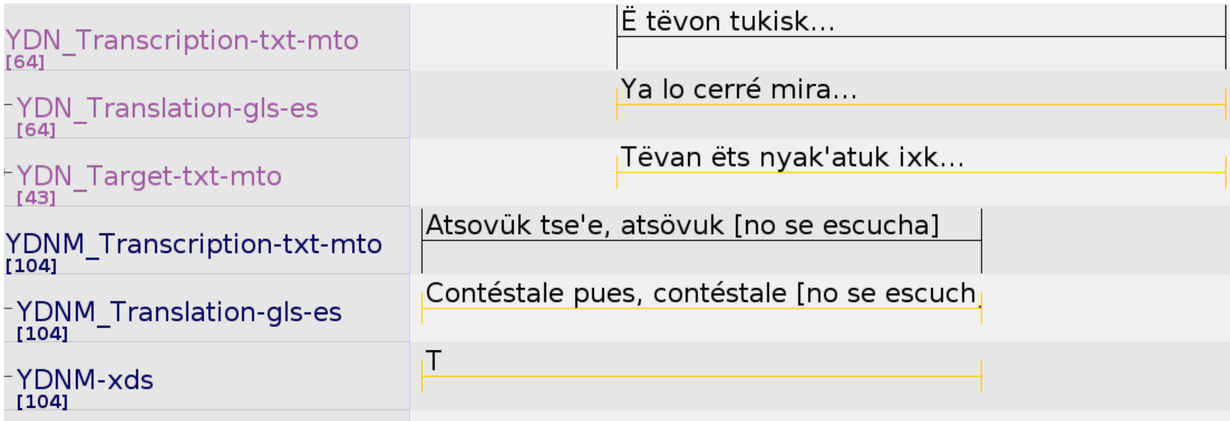

Figure 6. A portion of an ELAN (.eaf) transcript of child-caregiver interaction. Each turn is annotated for speaker and, in some cases, addressee; ungrammatical turns are annotated with their grammatical ("target") equivalent

As well as these annotation types, child language researchers often want to add morphological analyses to their transcripts. These analyses allow researchers to explore theoretical questions - for example, whether adults use less complex morphology while speaking to children - as well as curriculum-oriented and practical issues, such as identifying the nouns which children hear and use most frequently. Materials of this type can also feed into language documentation programs, where they may be combined with work on adult language use (Hellwig & Jung 2020), or in crosslinguistic projects, where the interlinear glossing is crucial.

3.3.1 Advantages of FLEx

For this use case, the greatest advantage of FLEx is its powerful parser. With large-scale collection of data on language use in interaction, researchers will likely need to analyze tens of thousands of turns at talk. This is time-intensive in any scenario, but can be done several times faster using a trainable tool like the FLEx XAmple parser. If the user has a digital lexicon of the language, they can speed up the process further by importing the lexicon to FLEx before analyzing the transcripts, which reduces the number of lexical entries they need to create during analysis. Furthermore, since their analyses will be linked to entries in the FLEx lexicon, they will have protection against glossing inconsistencies caused by typos or other human errors.

[9] Note that ungrammatical utterances are also frequently part of elicited adult language documentation corpora.

3.3.2 Disadvantages of FLEx

Import Process

FLEx and ELAN have limited interoperability, unless users either adapt them substantially or use a custom interchange format. Child language researchers encounter problems with FLEx when they import ELAN transcripts, during analysis, and again at export. To use the app's native import process, users need to alter their ELAN transcripts by (1) renaming types according to FLEx's required format; (2) renaming tiers; and (3) creating additional tiers that have no purpose outside of the FLEx import (e.g., a tier giving a text title). FLEx offers no built-in way to automate these time-consuming changes or to batch import EAFs. There is also no built-in CLAN import.

Suppression of Participant Information

When users view imported ELAN transcripts in the Texts & Words module of FLEx, they cannot see the speaker and addressee codes from the ELAN file, as shown in the screenshot in Figure 7. Speaker information is imported from ELAN on the FLExText backend – this is the "participant" attribute in the XML snippet shown in Figure 8, which is the source for Figure 7. But even though speaker information is present, FLEx does not display it to the user. Furthermore, addressee information is not imported at all. Suppressing this information makes conversation transcripts confusing and difficult to analyze, since users cannot tell if adjacent turns come from the same speaker or different speakers.

Suppressing participant information also makes the FLEx search feature almost completely unusable for many common child language research questions. For example, users cannot search only on child speech, or only on child-directed speech. To work around the suppression of participant information, researchers can add a speaker attribution, such as "Child1", or "Caregiver", to the beginning of each annotation in ELAN and then use the native import function of FLEx. The speaker attributions will then appear as part of the text baseline. This workaround is unsatisfactory: it adds even more time to the import process, and still does not allow users to view addressee information during analysis.

7 Word	Atsovük tse'e ,	atsövuk [	no	se	escucha]
Morphemes	***	***	no	***	***
Lex. Entries	***	***	***	***	***
Lex. Gloss	***	***	***	***	***
Lex. Gram. Info.	***	***	***	***	***
Word Gloss	***	***	no	***	***
Word Cat.	***	***	***	***	***

Free Spa Contéstale pues, contéstale [no se escucha]

Eng

8 Word	Tëvan	ëts	nyak'atuk	ixk...	
Morphemes	tëvan	ëts	***	ix	-k
Lex. Entries	tëvan	ëts	***	ix	$-k_2$
Lex. Gloss	already	PRO1.SG	***	look	IMP
Lex. Gram. Info.	adv	pro	***	v	v:Any
Word Gloss	already	PRO1.SG	***	look	
Word Cat.	adv	pro	***	v	

Free Spa Ya lo cerré mira...

Eng

Figure 7. A FLExText file as viewed in FLEx, where all information about participants, addressees, and timestamps has been suppressed

```
<phrase guid="00000000-0000-0000-0000-0000001028e0" begin-time-offset="13330" end-time-offset="15280" speaker="YDN"
media-file="00000000-0000-0000-0000-0000001027a5">
  <item type="segnum" lang="en">9</item>
  <words>
    <word guid="b02664f5-bb11-4701-a311-bf1f528fb471">
      <item type="txt" lang="cps">Ë</item>
      <morphemes>
        <morph type="stem" guid="d7f713e8-e8cf-11d3-9764-00c04f186933">
          <item type="txt" lang="cps">ëts</item>
          <item type="cf" lang="cps">ëts</item>
          <item type="gls" lang="en">PRO1.SG</item>
          <item type="msa" lang="en">pro</item>
```

Figure 8. The XML of the FLExText file shown in Figure 7: note that participant information and timestamps are retained from the original .eaf

An alternative is to import ELAN participant information as a note or (as of FLEx 9.1) a custom field on each line of the FLExText. The advantage of this strategy is that, because lines can have an unlimited number of notes, users can view both speaker and addressee information as separate notes. The disadvantage of using notes is that it requires circumventing the native import process, meaning that users must have the programming skills to create a custom ELAN-FLEx interchange format. Custom fields offer a lower-effort solution, but only for Windows users, since FLEx 9.1 is not currently available for Linux.

Search Performance

Even when researchers successfully use workarounds to add participant information, in practice they still cannot search only child or only adult turns in FLEx, because of the poor performance of the FLEx search features. Standard searches without regular expressions are feasible, but simple regex queries – for example, a regex search for the string "Child" followed by a specific morpheme – frequently crash the program or machine. Complex concordance searches cause the same problems. These search performance issues have existed since the first version of FLEx (Butler & van Volkinburg 2007; Rogers 2010). Furthermore, since none of the FLEx search functions allow users to save results to an external file, search results cannot easily be exported for statistical analysis.

Actual vs. Target Child Turns

If researchers have used the right EAF structure, during analysis they can import children's actual (potentially ungrammatical) turns, as well as their associated target (grammatical) turns as separate lines in the baseline. But because FLEx does not allow multiple surface-level representations of the same turn, the import will break the association between the actual and target turns. This means that researchers cannot search the FLEx database for any specific target turn-actual turn mapping – that is, they will not be able to search for any particular error type, such as omission of agreement, incorrect agreement, or overgeneralization.

As a solution to this issue, FLEx developer Beth Bryson suggests to us that researchers could enter target and actual forms as the same baseline line, using different writing systems for each form. This workaround is infeasible because it requires a one-to-one-mapping between words in the target and actual turns. This type of mapping is not usually present in child language, since children often delete obligatory elements or add ungrammatical ones.

Our own team's solution to the target-turns issue, discussed in [redacted for anonymous review], involves circumventing the native import/export process of FLEx so that target and actual turns are imported as separate baseline lines but associated via a unique note. Because of the poor performance of FLEx search, this still does not allow us to search only actual or only target lines in FLEx, or to systematically compare actual and target turns. Instead, we export the transcripts in FLExText XML format and read in the XML to other apps for analysis.

Ungrammatical Turns

An issue affecting both child language research and research based on grammaticality judgments is that the text module of FLEx does not include an option to mark lines as ungrammatical, nor to propose multiple possible analyses for any given word. Of course, researchers can use a symbol like '*' to indicate ungrammatical turns in the text baseline, but using these symbols will not exclude ungrammatical content from searches. To search only grammatical turns, researchers will again need to use the underperforming regex search.

Export Formats and Archiving

Given the poor performance of the FLEx search features, child language researchers need to export FLEx texts into another format for search when their analysis is complete. Exporting back to ELAN is potentially helpful, since it allows users to access speaker and addressee information again. But in practice, the native FLEx-to-ELAN export process is ridden with bugs. In ELAN files exported from FLEx, participant codes in tier names are replaced with arbitrary letters, and turns containing certain punctuation are broken into multiple phrases and attributed to spurious extra participants. Even more problematically, phrase-level transcriptions are erased and replaced by empty phrase-level annotations dominating a tokenized word tier, making it extremely cumbersome to search for multiword strings.

The loss of phrase-level transcription cannot be resolved by exporting to an alternative format. Phrase-level transcriptions are absent in the FLExText backend and therefore also absent in the app's other export formats, such as FLExText XML and XLingPaper. Researchers can work around this issue only if they have the technical skills necessary to edit the XML export by concatenating the text of the word nodes and copying it to the phrase nodes. Similarly, FLEx does not offer export to flat formats such as CSV. Users who need CSVs for statistical analyses must export to one of the XML formats offered by FLEx, and then parse the XML in another app.

Since researchers are unlikely to archive the poorly structured ELAN files exported from FLEx, they may decide to deposit their entire FLEx database in the archive instead. This will require them to decide between archiving individual XML files in FLExText or XLingPaper format, or archiving a backup of the entire database, with the extension .fwbackup. The XML files are more human-readable and much more obviously related to the texts, but they do not include the project lexicon and therefore <u>cannot</u> be used to restore the database. Restoration is possible only with the entire .fwbackup file, making it very easy for users to unwittingly archive unusable materials.

3.3.3 Implications

With these attributes, FLEx encourages certain types of research and discourages others. These are summarized in Table 5.

Table 5. Implications of FLEx data structures for transcript analysis

Attribute	Encourages	Discourages
Import of time-aligned files from ELAN is cumbersome	Analyzing only born-text materials	Analyzing audio or video transcripts
Text module suppresses speaker and addressee information	Analyzing only monologues	Studying conversation Comparing speech by different types of speakers, e.g., learners vs. fluent speakers Comparing speech addressed

		to different types of listeners, e.g., to children vs. to adults
Regex searches and complex concordance feature crash machine or program	Using simple concordance as only search tool	Using all available search features
Search results cannot be exported to CSV	Manually entering search results into other software for analysis	Creating fully reproducible, dynamically updating analysis pipeline (à la R Markdown)
ELAN export is poorly structured	Using FLEx as endpoint of analysis	Re-exporting to ELAN after analysis
Exports are in formats not commonly used by linguists or communities, e.g., no flat formats, no phrase-level transcription	Using FLEx as endpoint of analysis	Using FLEx as one element of pipeline for later analyses with, e.g., R

3.4 Historical Text Analysis

Previous sections have focused on language data gathered directly with research participants. Language documentation projects also involve work with written materials collected by other linguists. In this section, we focus on issues related to work with secondary text collections; that is, those created as part of "salvage" linguistics and Boasian documentation projects (cf. Musgrave & Thieberger 2021). Such projects involve analyzing and interpreting older sources for reclamation, historical interest, or other analytical research. Recent examples include Begay (2017) for Wailaki and Brixey and Artstein (2021) for Choctaw. Lukaniec (2021) gives an overview of these processes with examples from Wendat.

This type of work should be highly suited to FLEx, since the primary aim of such a project is the transcription, translation, and analysis of textual material. However, even fairly simple historical text collections raise problems for FLEx's data structures, and the special requirements for linguistic work – particularly morphological parsing and linguistic annotation – mean that software developed for other historical or annotative work is inappropriate.[10] The problem can be summarized as follows: working with historical texts involves intertwined presentation and analysis of primary data. FLEx makes it difficult to keep track of decisions, because its data structures require knowing things that cannot be known until the analysis is done. The solutions

[10] We have in mind software such as Mirador (for image collation and annotation) or NVivo (for document annotation and knowledge graph creation).

to these problems look a lot like the data structures that are required for work on other formats, as described above.

Working with textual collections from different sources and periods introduces additional required metadata. In addition to the contributor, situational, and item format metadata that Sullivant (2020) discusses, historical documents often have format metadata, such as page layout information.[11] Anderson (2020) gives examples from paradigms in a Tunica manuscript. Contributor data is particularly important for studying variation among speakers and for properly attributing individual or family intellectual property (cf. Lukaniec 2022). Such metadata may need to be associated with items at the word level, as well as to a text as a whole.

Historical text collections often contain materials from different periods, authors, and transcribers. The source materials often use varied, usually poorly documented, transcription conventions, both within the collection and compared to standards used within the contemporary speech community. Researchers usually want to keep track of these conventions and not standardize them immediately, as the process of conducting research with the text collection may change their knowledge of how the conventions work.[12] Some of these conventions might be "surface" (that is, simply convertible, such as using <ṅ> or <ng> for /ŋ/). Others may be more embedded in the media, such as using spatial layout on the page to signal language, as in the Bates collection[13] typescripts.

FLEx handles such inputs through the twin concepts of "language" and "writing system". That is, a project may include multiple languages (object language, analysis language, and lingua francas, for example) and within a language there may be multiple writing systems (practical orthography, IPA, syllabary, etc.).[14] Some types of material fit easily in this dichotomy. Standard Uzbek Cyrilic vs. Standard Uzbek romanization, for example, has straightforward mappings between character sets. When dealing with multiple different inputs for older materials, however, the breakdown is not as straightforward. That is, textual differences might not be purely spelling variants (though they may include spelling variants); they aren't different writing systems (though they may be); and they aren't different languages or varieties (though they may be). They are different doculects (Cysouw & Good 2013) with different conventions of representation, all of which need to be mapped onto the standards for the documentation project.

Such projects typically have interleaved and complex annotation types. For the Laves Bardi text collection, for example, we have Laves' original annotations, our comments on Laves' texts and

[11] Paradigms, for example, contain information about word relationships that are represented spatially.

[12] For example, fieldnotes from archival sources may not be explicit about transcription conventions, and only subsequent research, enabled by the corpus, might lead to an understanding of the conventions used.

[13] http://bates.org.au/text/

[14] Note that FLEx also treats audio input as a "writing system", but we do not discuss that further here.

translations, our comments on his annotations, and meta-commentary (Bowern 2003). That is, there is information provided by Laves, information provided by Bardi elders on the original speakers' materials and Laves' comments, and Bowern's annotations that link to other materials in the collection and clarify grammatical and lexical concepts. These are annotations at the word, phrase, sentence, and annotation block level.

Finally, such projects may have associated audio or video files, but typically it is not the audio recordings of the original texts (as in the previous use cases described above). There might be audio associated with the analysis of the print materials which we should link to the texts, but these recordings are not direct representations of the texts. For example, Katie Bicevskis's dissertation involved archival work with Ian Green's recordings of Marri Ngarr (Paradisec collection IG3; https://catalog.paradisec.org.au/collections/IG3/); her fieldwork involved checking and discussing the original materials.[15]

3.4.1 Advantages of FLEx

FLEx has many advantages for historical data projects. The greatest is the ability to integrate materials with a lexicon – either via an existing dictionary, or by organizing forms compiled through the texts to create a dictionary. FLEx's ability to gloss at the word level as well as and parse through a fairly flexible morphological parser (with morphological segmentation) allows for schematic glossing and collection of examples before the morphology is fully known. It allows for preliminary annotation and subsequent analysis as knowledge of the language increases, meaning that a text collection may be developed incrementally. Files can be backed up at waypoints and exported. The linking of dictionary and text allows examples to be selected from the text corpus.

3.4.2 Disadvantages of FLEx

Historical text collections also have some of the problems that are discussed in §3.3.2 on child language. The contents of the "text" line in this data are not uniform – need not be grammatical, need not be in the same language, etc. But FLEx assumes that lines are uniform, and this assumption – along with the lack of word-level annotation functionality in text views – makes it difficult or impossible to track the type of information that the researcher needs.

It is difficult to export material from FLEx. The export processes for textual materials are limited and many do not preserve the very information that makes FLEx a logical choice for such projects. This includes the spacing of interlinear glossed text, which must be either copied-and-

[15] See further https://www.paradisec.org.au/blog/2023/04/grammars-from-archival-records/ for discussion of this methodology.

pasted using a fixed width font or exported only to SIL's own text editor, or the interlinear tabbing must be manually recreated.[16]

FLEx's search functions also limit historical text use. While it is possible to specify a genre in the metadata of a text, this information is not exported in the FLExText output. Likewise, text dates can be specified but not used to sort or filter. In effect, it is possible to enter metadata about the text but impossible to use that information when searching or exporting the corpus (this parallels the search issues discussed in §3.3.2). The ability to sort and search by date, as well as by genre, is particularly important for text collections that span centuries.

3.4.3 Workarounds

Previous FLEx projects have worked around these issues in a few different ways. One option is to standardize the input (e.g., Bowern's 2003 Laves project with Bardi standardized all spelling to the 1990 Bardi orthography). Another option is to treat all variants as "spelling variants" (or define another variant type in FLEx). Export workarounds include exporting texts to HTML, or using ODT (OpenDocument Text format) in software such as LibreOffice, which FLEx supports but which can cause problems in subsequent parts of the workflow. Other export pipelines (such as HTML > PDF > Word) keep the text but not its spacing; in such cases the interlinear spacing is recreated through borderless tables or regular tab placement. Thus, in summary, for text exports the options are either to change other parts of the workflow to accommodate FLEx, or to recreate interlinear alignment.[17]

3.4.4 Implications

FLEx encourages certain metadata and annotation practices in historical text work and discourages others. Table 6 summarizes these issues.

Table 6. Implications of data structures for historical text work

Attribute	Encourages	Discourages
Must associate all language input with a "Language" and "Writing System"	Irreversible decisions before analysis has been undertaken	Waiting until later in analysis to assign text to language or orthography
Metadata	Structured metadata (an advantage)	Metadata that is needed for the types of texts used

[16] In theory, it is possible to export from the export menu to an XML format that is readable in Microsoft Word. However, in our experience, this either produces a corrupt or unopenable file, or a file that treats every line of interlinear text as an equation. In one export, the file resulted in tables where the cell width was the width of the page, so every sentence had a single morpheme per line.
[17] LaTeX users will point out that a FLEx > LaTeX pipeline has no need to retain tab spacing, but this brings additional issues. Not least, LaTeX is hardly used outside of academia, restricting who can edit materials.

Simple annotation	Minimal annotation, annotation at the word level (applicable to the lexical item, not the instance)	Annotation at level of single word token
Export destroys interlinear alignments	Disseminating only baseline text	Exporting text collections for sharing

4 General Discussion

Since language workers have many varied needs, having a centralized technological toolkit for managing a fieldwork project is a major asset. The cases described above all demonstrate the problems that can arise when users choose a multi-purpose tool to satisfy relatively more specialized project needs.

While users may be aware of these problems ahead of time, they continue to make this choice because FLEx is a salient option for language workers, and often arises as the primary suggestion for database management software. FLEx provides solutions for keeping together the entire Boasian trilogy–corpus, lexicon, and grammar. It is the only language documentation tool that integrates a parser. As a result, items in each module can be linked to each other, and FLEx delivers handily on its promise to help with glossing texts by using the lexicon. With various import and export options, users can integrate data from other sources, and also prepare it for use elsewhere. In the lexicon, users can add audio and image data and specify variants, multiple senses, and allomorphs. In the corpus, users can perform regular expression concordance searches, automatically gloss text, and add lexemes from within the text view. In the grammar module, users can add phonological rules that help the parser make hypotheses about what might be an allomorph, as well as take notes on grammatical features as the user learns more about the language of study.

FLEx delivers on these promises, but there are many ways in which it fails to deliver on the expectations that users have when adopting it as their database management tool of choice. As illustrated in the case studies above, FLEx makes many assumptions about not only what language work looks like, but how language works. Through the program's lifetime, the developers of FLEx have recognized and attempted to mitigate some of these assumptions. Though the software continues to miss the mark in some ways, these incremental improvements cause users to retain loyalty in hopes that the program will improve with time. After all, because FLEx lacks viable competitors, users' only options are either to create workarounds or to resign themselves to working with only the types of data that FLEx is optimized to analyze.

4.1 Assumptions About Language and Data Structure

FLEx is optimized for language data that is *mono-*: monolingual, monolectal, monologic, and monomodal. It works best when the data is homogenous - for example, always grammatical and always in the same consistent orthography - and the parser functions best on exclusively concatenative morphology. This optimization reflects a set of implicit assumptions about what languages and types of data fieldworkers want to analyze. When we analyze only homogenous and “mono” data, these assumptions are not obvious. But when we analyze data that departs from any of the assumptions, they become clear immediately.

Secondly, FLEx’s assumptions about language and data structure raise unnecessary barriers to many types of research. The parser’s inability to analyze non-concatenative morphology adds unnecessary time and labor cost to morphological research. Its assumption that all lines of text are monolingual does the same for work on code-switching and naturalistic speech in multilingual communities. Similarly, the assumption that all lines are grammatical means that, without extensive adaptation, the program is unsuitable for work on child language or with elicited judgment data. While these are the issues that our case studies discuss in depth, they are hardly the only assumptions that FLEx makes. Further problems include that FLEx does not support video media, meaning that it cannot be used to analyze signed languages,[18] and that it requires all annotations to correspond to a single line of the baseline text, which prevents users from adding information about prosody, gesture, co-occurring activity, or other behaviors that have different time boundaries from the baseline text lines.

With these assumptions, FLEx encourages the user either to rely only on “mono” data or to “clean up” more complex text into mono form, following much the same research practices that Boas and the early Americanists used more than a century ago (Epps, Webster & Woodbury 2017: 45). While research based on mono materials has a long and important history in language documentation, it is not the only way to document a language, and it is not sufficient for many research questions. The field needs tools that are capable of analyzing topics – child language, signed languages, multilingualism – that contemporary researchers and revitalization practitioners have highlighted as documentation priorities.

4.2 Assumptions About the User

As well as being built for a specific type of language data, FLEx is also built for a specific type of user – an academic linguist or a missionary with some training in linguistics. While other SIL software products, such as WeSay, are intended for a broader user base,[19] FLEx clearly is not.

[18] SIL does, however, offer SooSL, a dedicated software program for assembling signed-language dictionaries (https://soosl.net/). It has a MacOS version, but this consists only of the dictionary component and does not include text collections or a parser.

[19] Per https://software.sil.org/wesay/faq/, WeSay is not designed as a standalone program, but in conjunction with a “normal [sic]” computer running FLEx, to complete additional tasks.

The software's technical (e.g. OS and RAM) requirements tend to exclude community-insider language workers from using it. Nevertheless, even language workers with access to FLEx are likely to have a poor user experience: the interface is unwieldy, the learning curve is steep, and the collaboration features are difficult to use. Once analysis in FLEx is complete, the outputs are not customizable to community needs. As a result, dependence on FLEx tends to both exclude community-insider language workers from key parts of analysis and promote workflows that are not conducive to their goals.

4.3 Attempted Solutions by FLEx Developers

Over the lifespan of FLEx, the program's developers have tried to remedy some of the most common user complaints. However, these changes still do not meet the needs of many users. For example, the developers expanded the OS compatibility of FLEx to Linux in addition to Windows, but have not expanded to MacOS, despite the high demand for that feature. Similarly, as of FLEx 7.3, users can collaborate on a database with people using other computers, but collaboration still requires a finicky, opaque version control process.

Beyond these changes to the technical requirements, recent changes to FLEx have also responded to analysis- and publication-related complaints about the program. These changes, however, are also less than satisfactory. For example, previous versions of FLEx also assumed that the outputs of the lexicon and corpus could always take the same form. Over the last several versions, the program's developers have acknowledged that users need to create different kinds of resources from FLEx databases, and they have added compatibility with more programs. Yet customization of dictionaries and texts is still extremely limited, and the output file formats are still not conducive to average users modifying either the aesthetics or the content of outputs. In the same vein, as of FLEx 9.1, users can add custom fields to each line of text. Custom fields allow users to organize various forms of line-level annotations that were more difficult to organize in previous versions of FLEx, but they still make the assumption that all annotations should appear at the utterance level, rather than more or less granular levels, such as word or paragraph.

4.4 Attempted Solutions by Users: A Culture of Workarounds

Despite the changes to the software outlined above, there is a persistent mismatch between researchers' goals and FLEx's functionality. In response to this, the FLEx user community has developed a pervasive culture of workarounds. Some of these workarounds are small tweaks within the database, such as creating an 'EnglishWord' headword to annotate instances of code-switching. But other workarounds, like building custom converters to move data between FLEx and other apps (Beier & Michael 2022), require programming knowledge – meaning that teams must either recruit programmers or upskill on their own in order to create working dataflows.

This need for workarounds, whether small or large, creates serious bottlenecks. At best, it slows the progress of language work. At worst, when teams lack the resources to create a workaround, they may abandon FLEx and switch to an inefficient strategy such as glossing text entirely by hand. Furthermore, as well as being costly in time and money, workarounds are fragile. They have to assume particular XML structures in the input and output of FLEx, and this means they are liable to break when the app is updated.

Other workarounds create problems for archiving and future analysis. FLEx builds numerous linguistic structures into the database (roots, stems, clitics, derivational affixes, for example) as preset options. This is intended to speed up analysis. However, because those items are associated with particular parsing behavior, users are encouraged to pick a category that produces a particular type of parse behavior, even if it results in a specific claim about grammar. A June 20, 2023 post to the FLEx-List Google group, for example, encouraged a workaround where absolutive case affixes were treated as derivational morphology. Such choices have implications for future grammar analysts.

5 Conclusions

As a discipline, language documentation relies on old software. Many of these older digital tools are still functional. Though they may not encourage collecting new forms of data, they also do not prevent it – for example, ELAN is not designed for work with children, but its data structures are flexible enough that it is widely used in child language research nevertheless. Other legacy tools, however, have negative impacts on the field because of their monopoly status and rigidity. FLEx is the prime example here: through all of the data and analysis structures that we have described above, the software encourages users to retain historic methods and raises barriers to innovative types of research.

In sum, we are faced with an infrastructure problem for the discipline. Fieldworkers are traveling the same (or similar) roads and are encountering the same barriers. As we have shown in the previous sections, teams are mostly building their own detours around these barriers – creating workarounds – rather than repairing the shared road. Going forward, the discipline will need to reflect on the consequences of upholding this status quo. This reflection offers us an opportunity to collectively construct our future relationship with digital infrastructure. To do so, documentation researchers will need a centralized space to discuss technological problems that arise and ideas about potential solutions. Many of today's ad hoc solutions could apply to projects other than the ones that they were created to support, yet this potential is not often realized, whether because the solutions are idiosyncratic or because they never circulate beyond a single research team.

In contrast to the status quo, language documentation needs a digital infrastructure that is flexible and generalizable enough to accommodate teams' diverse goals, questions, and workflows. This infrastructure also needs to be sustainable, in two ways – it needs institutional support, in the form of funding, updates and technical support, and it needs a user community where people can exchange solutions to common problems, rather than duplicating efforts in isolation. These forms of support are what have made FLEx the standard, but they do not need to be unique to FLEx. Instead, our goal is to begin a conversation about how the field can create digital infrastructure that is shared, supported, and flexible.

6 References

Anderson, Patricia. 2020. *Revitalization Lexicography: The Making of the New Tunica Dictionary*. University of Arizona Press.

Anthony, Laurence. 2022. AntConc. Tokyo: Waseda University. https://www.laurenceanthony.net/software.

Baldwin, Daryl, David Costa & Douglas Troy. 2016. Myaamiaataweenki eekincikoonihkiinki eeyoonki aapisaataweenki: A Miami Language Digital Tool for Language Reclamation. *Language Documentation & Conservation* 10. 394–410. http://hdl.handle.net/10125/24713.

Beier, Christine & Lev Michael. 2022. Managing Lexicography Data: A Practical, Principled Approach Using FLEx (FieldWorks Language Explorer). In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 301–314. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0029.

Bird, Steven & Gary Simons. 2003. Seven Dimensions of Portability for Language Documentation and Description. *Language* 79(3). 557–582. https://doi.org/10.1353/lan.2003.0149.

Black, Andy, Gary Simons & Ken Zook. 2007. *Parser and interlinear text*. https://software.sil.org/fieldworks/wp-content/uploads/sites/38/2016/10/Parser-and-interlinear-text.pdf.

Boersma, Paul & David Weenink. 2022. Praat: doing phonetics by computer. http://www.praat.org.

Bowern, Claire. 2003. Laves' Bardi texts. In Joseph Blythe & R. McKenna Brown (eds.), *Maintaining the links: Language, identity and the land. Proceedings of the Seventh FEL [Foundation for Endangered Languages] Conference*, 137–143. Broome: Foundation for Endangered Languages.

Bowern, Claire. 2015. *Linguistic fieldwork: practical guide*. Second Edition. New York: Palgave Macmillian.

Buckskin, Jack Kanya, Taylor Tipu Power-Smith, Jaylon Pila Newchurch, Tempestt Sumner-Lovett, Paul Finlay, Chester Schultz & Rob Amery. 2021. Innovative strategies for reintroducing a sleeping language: How a community-university partnership supports the revitalization of Kaurna, the language of the Adelaide Plains, South Australia. *WINHEC: International Journal of Indigenous Education Scholarship* (1). 47–90.

Butler, Lynnika & Heather van Volkinburg. 2007. Review of Fieldworks Language Explorer (FLEx). *Language Documentation & Conservation* 1. 100–106. http://hdl.handle.net/10125/1730.

Caballero, Gabriela. 2017. Choguita Rarámuri (Tarahumara) language description and documentation: a guide to the deposited collection and associated materials. *Language*

Documentation and Conservation 11. 224–255.
Chelliah, Shobhana Lakshmi & Willem Joseph de Reuse. 2011. *Handbook of descriptive linguistic fieldwork*. Dordrecht [Netherlands] ; New York: Springer.
Chew, Kari A.B., Lokosh & Juliet Morgan. 2022. Centering relationality in online Indigenous language learning: Reflecting on the creation and use of Rosetta Stone Chickasaw. *Language Documentation & Conservation* 16. 228–258. http://hdl.handle.net/10125/74679.
Chiblow, Susan & Paul J. Meighan. 2022. Language is land, land is language: The importance of Indigenous languages. *Human Geography* 15(2). 206–210. https://doi.org/10.1177/19427786211022899.
Cox, Christopher. 2022. Managing Data in a Language Documentation Corpus. In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 277–286. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0027.
Cysouw, Michael & Jeff Good. 2013. Languoid, doculect and glossonym: Formalizing the notion "language." *Language Documentation & Conservation* 7. 331–359. http://hdl.handle.net/10125/4606.
Davis, Jenny L. 2017. Resisting rhetorics of language endangerment: Reclamation through Indigenous language survivance. *Language Documentation and Description*. Language Documentation and Description 37-58 Pages. https://doi.org/10.25894/LDD147.
Di Carlo, Pierpaolo, Rachel A. Ojong Diba & Jeff Good. 2021. Towards a coherent methodology for the documentation of small-scale multilingualism: Dealing with speech data. *International Journal of Bilingualism* 25(4). 860–877. https://doi.org/10.1177/13670069211023144.
Dobrin, Lise & Jeff Good. 2009. Practical Language Development: Whose Mission? *Language* 85(3). 619–629.
Dunham, Joel, Gina Cook & Joshua Horner. 2014. LingSync & the Online Linguistic Database: New Models for the Collection and Management of Data for Language Communities, Linguists and Language Learners. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, 24–33. Baltimore, Maryland, USA: Association for Computational Linguistics. https://doi.org/10.3115/v1/W14-2204.
Epps, Patience L., Anthony K. Webster, & Anthony C. Woodbury. A Holistic Humanities of Speaking: Franz Boas and the Continuing Centrality of Texts. *International Journal of American Linguistics* 83(1). 41–78. https://doi.org/10.1086/689547.
Gawne, Lauren. 2018. A Guide to the Syuba (Kagate) Language Documentation Corpus. *Language Documentation & Conservation* 12. 204–234. http://hdl.handle.net/10125/24768.
Gonzalez, J. L., Anuschka Van'T Hooft, Jesus Carretero & Victor J. Sosa-Sosa. 2017. Nenek: a cloud-based collaboration platform for the management of Amerindian language resources. *Language Resources and Evaluation* 51(4). 897–925. https://doi.org/10.1007/s10579-016-9361-8.
Grassroots Indigenous Multimedia. 2023. Reclaim! https://gim-ojibwe.org/reclaim.
Grzech, Karolina & Selena Tisalema Shaca. 2022. Building trust on Zoom: A workflow for language documentation via videoconferencing software. *Language Documentation &*

Conservation 16. 79–97. http://hdl.handle.net/10125/74673.
Harvey, Dan. 2022. ACORNS: Acquisition of Restored Native Speech. http://cs.sou.edu/~harveyd/acorns/index.php
Hatton, John, Gary Holton, Mandana Seyfeddinipur & Nick Thieberger. 2021. Lameta. https://github.com/onset/laMETA/releases.
Hellwig, Birgit & Dagmar Jung. 2020. Child-directed language – and how it informs the documentation and description of the adult language. *Language Documentation & Conservation* 14. 188–214.
Hermes, Mary. 2012. Indigenous Language Revitalization and Documentation in the United States: Collaboration Despite Colonialism: Collaboration Despite Colonialism. *Language and Linguistics Compass* 6(3). 131–142. https://doi.org/10.1002/lnc3.327.
Hochgesang, Julie A. 2022. Managing Sign Language Acquisition Video Data: A Personal Journey in the Organization and Representation of Signed Data. In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 367–384. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0035.
Indigenous Protocol and Artificial Intelligence Working Group. 2020. Guidelines for Indigenous-centred AI design v. 1. In Jason Edward Lewis (ed.), *Indigenous Protocol and Artificial Intelligence Position Paper*, 20–24. Honolulu, Hawai'i: The Initiative for Indigenous Futures and the Canadian Institute for Advanced Research (CIFAR).
Leonard, Wesley Y. 2017. Producing language reclamation by decolonising 'language.' *Language Documentation and Description*. Language Documentation and Description Vol. 14 (2017). https://doi.org/10.25894/LDD146.
Lukaniec, Megan. 2022. Managing Data from Archival Documentation for Language Reclamation. In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 315–326. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0030.
MacWhinney, Brian. 2000. *The CHILDES project: tools for analyzing talk*. 3rd ed. Mahwah, NJ: Lawrence Erlbaum.
Mattern, Eleanor. 2022. The Linguistic Data Life Cycle, Sustainability of Data, and Principles of Solid Data Management. In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 61–72. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0009.
Maxwell, Michael. 2003. *Hermit Crab: parsing and generating with classical generative phonology and morphology*. Dallas: SIL International Digital Resources. https://www.sil.org/resources/archives/5746.
McCarty, Teresa L., Sheilah E. Nicholas, Kari A. B. Chew, Natalie G. Diaz, Wesley Y. Leonard & Louellyn White. 2018. Hear Our Languages, Hear Our Voices: Storywork as Theory and Praxis in Indigenous-Language Reclamation. *Daedalus* 147(2). 160–172. https://doi.org/10.1162/DAED_a_00499.
Meakins, Felicity, Jennifer Green & Myfany Turpin. 2018. *Understanding linguistic fieldwork*. London: Routledge.
Miromaa Aboriginal Language & Technology Centre. 2023. Miromaa. https://www.miromaa.org.au/miromaa-software/get-started.
Musgrave, Simon & Nick Thieberger. 2021. The language documentation quartet. In *Proceedings of the 4th Workshop on the Use of Computational Methods in the Study of*

Endangered Languages Volume 1 (Papers), 6–12. Online: Association for Computational Linguistics. https://aclanthology.org/2021.computel-1.2.
Palfreyman, Nick. 2022. Managing Sign Language Data from Fieldwork. In Andrea L. Berez-Kroeker, Bradley McDonnell, Eve Koller & Lauren B. Collister (eds.), *The Open Handbook of Linguistic Data Management*, 267–276. The MIT Press. https://doi.org/10.7551/mitpress/12200.003.0026.
Rehm, Georg. 2016. The Language Resource Life Cycle: Towards a Generic Model for Creating, Maintaining, Using and Distributing Language Resources. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2450–2454. Portorož, Slovenia: European Language Resources Association (ELRA). https://aclanthology.org/L16-1388.
Rogers, Chris. 2010. Review of FieldWorks Language Explorer (FLEx) 3.0. *Language Documentation & Conservation* 4. 78–84.
Sande, Hannah & Marcus Ewert. 2023. TwistedTongues. https://twisted-tongues-beta.appspot.com/#/.
Silva, Wilson de Lima. 2016. Animating Traditional Amazonian Storytelling: New Methods and Lessons from the Field. *Language Documentation & Conservation* 10. 480–296. http://hdl.handle.net/10125/24703.
Skilton, Amalia. 2021. Ticuna (tca) language documentation: A guide to materials in the California Language Archive. *Language Documentation & Conservation* 15. 153–189. http://hdl.handle.net/10125/24972.
Sullivant, Ryan. 2020. Archival description for language documentation collections. *Language Documentation and Conservation* 14. 520–578.
Summer Institute of Linguistics. 2022. FieldWorks Language Explorer. Version 9.1.18. https://software.sil.org/fieldworks/download/fw-91/fw-9118/.
Warner, Natasha, Quirina Luna & Lynnika Butler. 2007. Ethics and revitalization of dormant languages. *Language Documentation & Conservation* 1(1). 58–76. http://hdl.handle.net/10125/1727.
Whalen, D. H., Melissa E. Lewis, Stefanie Gillson, Brittany McBeath, Bri Alexander & Kate Nyhan. 2022. Health effects of Indigenous language use and revitalization: a realist review. *International Journal for Equity in Health* 21(169).
Williams, Nicholas, Wilson de Lima Silva, Laura McPherson & Jeff Good. 2021. COVID-19 and documentary linguistics: Some ways forward. *Language Documentation and Description*. Language Documentation and Description 359-377 Pages. https://doi.org/10.25894/LDD57.
Wittenburg, Peter, Hennie Brugman, Albert Russel, Alex Klassmann & Han Sloetjes. 2006. ELAN: a professional framework for multimodality research. In *5th International Conference on Language Resources and Evaluation (LREC 2006)*, 1556–1559.
Woods, Lesley. 2023. *Something's Gotta Change: Redefining Collaborative Linguistic Research*. 1st edn. ANU Press. https://doi.org/10.22459/SGC.2022.