LingSync & the Online Linguistic Database: New models for the collection and management of data for language communities, linguists and language learners

Joel Dunham
University of British Columbia,
Department of Linguistics
jrwdunham@gmail.com

Gina Cook
iLanguage Lab
Montréal
gina.c.cook@gmail.com

Joshua Horner
Amilia
Montréal
josh.horner@gmail.com

Abstract

LingSync and the Online Linguistic Database (OLD) are new models for the collection and management of data in endangered language settings. The LingSync and OLD projects seek to close a feedback loop between field linguists, language communities, software developers, and computational linguists by creating web services and user interfaces (UIs) which facilitate collaborative and inclusive language documentation. This paper presents the architectures of these tools and the resources generated thus far. We also briefly discuss some of the features of the systems which are particularly helpful to endangered languages fieldwork and which should also be of interest to computational linguists, these being a service that automates the identification of utterances within audio/video, another that automates the alignment of audio recordings and transcriptions, and a number of services that automate the morphological parsing task. The paper discusses the requirements of software used for endangered language documentation, and presents novel data which demonstrates that users are actively seeking alternatives despite existing software.

1 Introduction

In this paper we argue that the LingSync/OLD project is a sustainable new model for data management which facilitates a feedback loop between fieldworkers, language communities, computational linguists, and software developers, thereby improving the effectiveness of language documentation efforts for low-resource language communities. In §2.1 we present five requirements for endangered languages fieldwork software which are currently not met by existing tools, as discussed in §2.2. Architectural considerations[1] under LingSync and the OLD which address these requirements are briefly outlined in §3. The ability of LingSync/OLD to integrate with existing software libraries commonly used in language documentation projects is demonstrated in §5. Finally, §6 demonstrates how the LingSync/OLD project is already seeing some closure of the feedback loop both in creating language learning apps for heritage speakers and in training Kartuli speakers to build speech recognition systems built on LingSync/OLD data.

2 Endangered languages fieldwork

Endangered languages are valuable culturally and scientifically, to their communities of origin (Ironstrack, 2012) and to humanity as a whole (Harrison, 2007). Efforts must be made to document these languages while there is still time (Good, 2012a; Thieberger, 2012). In cases where there are no longer any native speakers, a community may embark upon a language reclamation project that is wholly dependent upon the the products of past language documentation efforts (Leonard, 2012; Costa, 2012). Alongside such documentation and revitalization/reclamation projects is research-driven linguistic fieldwork. These diversely motivated yet interconnected strands within endangered languages fieldwork conspire to produce a particular set of requirements for effective software in this domain.

2.1 Software requirements

The following five requirements are essential, we claim, to effective language documentation soft-

[1] For further discussion of actual user interaction, screenshots and how LingSync/OLD data can be exported/published in existing online linguistics repositories such as EOPAS http://www.eopas.org/ and OLAC http://www.language-archives.org/ see Cathcart et al. (2012).

Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages, pages 24–33,
Baltimore, Maryland, USA, 26 June 2014. ©2014 Association for Computational Linguistics

ware: *integration of primary data*, *curation of data*, *inclusion of stakeholders*, *openable data*, and *user productivity*.

Requirement 1 *Integration of primary data*

While language reclamation projects founded solely on textual data can achieve some degree of success (Ironstrack, 2012), primary audio/video data *in the form of engaging content* is crucial to fostering native-like proficiency. Primary audio has formed part of language documentation efforts since the days of phonographs, yet only rarely have such audio products been made accessible. Securely and efficiently supporting the integration of primary audio/video data with text artifacts (e.g., dictionaries, grammars, collections of narratives) is part of the requirements of any modern language documentation effort (Schroeter and Thieberger, 2006; Good, 2012b).[2]

Requirement 2 *Curation of data*

While most language documentation literature places emphasis on the creation of publishable artifacts, our experience has shown that a significant percentage of language documentation hours are actually dedicated to the curation and filtering of the data in preparation for publication.[3] Even "a funding body like the ELDP cannot get all of its grantees [only 110 out of 216] to deposit in an archive in a timely fashion (or at all)" (Thieberger, 2012). We argue that facilitating the collaborative curation of data is, in fact, a core requirement of any data management or content management software, one which is largely overlooked by existing software (cf. §2.2).

Requirement 3 *Inclusion of stakeholders*

A sustainable language documentation effort involves crucially the creation of a positive feedback loop where the outputs of certain activities fuel the advancement of others. However, realizing this feedback loop requires tools that facilitate the inclusion of the various stakeholders involved in the process of language documentation *while* a project is underway, not *post hoc* when the data is "polished," which in 50% of projects never happens (Thieberger, 2012). This inclusivity requirement means that data and data processes must be available in formats that are usable to both humans—i.e., via graphical user interfaces (GUIs)—and machines—i.e., via software libraries and application programming interfaces (APIs).

Requirement 4 *Openable data*

One of the unique challenges associated with endangered languages fieldwork is the possibility that speakers or language communities may require that all or aspects of the raw data be kept confidential for a certain period of time.[4] Labs looking to reuse the data collected by field teams may, in particular, be unaware of the post-colonial context in which many fieldwork situations are embedded.

In the field it often happens that a speaker will speak quite candidly or receive a phone call during a recorded elicitation session and may want to restrict access to all or parts of that recording for personal reasons.[5] In some cases the living speakers of the language are so few that even anonymizing the data does not conceal the identity of the speaker from other speakers in the community. It also happens that particular stories or descriptions of rituals and cultural practices may need to be restricted to just the language community or even to sub-groups within the community.[6]

In order to provide access to all team members and stakeholders (including stakeholders who are distrustful of the project) language documentation software must support a non-trivial permissions system while also facilitating transparency

[2] For a more detailed discussion of the technical limitations which are no longer blocking the implementation of these requirements see Cathcart et al. (2012).

[3] Such artifacts might include engaging content to be reused in revitalization efforts, or citable/traceable data sets used to support research claims.

[4] Outside of language documentation contexts there are numerous valid reasons for facilitating data privacy. As with social websites (Facebook, YouTube), user data is generally considered private and not accessible to data scientists. Many content curation sites (Google Docs, WordPress) allow for content that is private indefinitely or during a pre-publication stage.

[5] Of course, as one reviewer points out, basing claims on private data runs contrary to a core tenet of the scientific method, namely that claims must be able to be assessed with transparent access to the methods and data used to support them. However, in these contexts field linguists generally protect the privacy of their language consultants by eliciting novel sentences which have similar grammatical features for publication, rather than using the original narrative. In the contexts of open data, such highly personal sections of transcripts must be "blacked out" so that the majority of the data can be made open.

[6] It is highly preferable for language communities to produce their own content using YouTube and other content sites, permitting the community to manage censorship of sensitive topics and personal narratives while creating more public data.

and encouraging open collaboration. Even language documentation projects using ad hoc content creation solutions (discussed in §2.2) cannot be fully inclusive for fear that when speakers of different dialects disagree they will "correct" each other's data if neither social pressure nor the permissions system prevents it. In fact, disagreements about data judgments remain an untapped indirect source of grammaticality information for linguistics researchers as there are no language documentation systems which permit inclusion of all stakeholders via traceable user activity, non-trivial permissions systems, and confidentiality of attributes on data. While not all teams will resort to data encryption or private data, implementing these features permits more stakeholders to have direct conditional access to data and removes barriers to adoption by language communities who may be initially distrustful of language documentation projects.

Requirement 5 *User productivity*

Users are accustomed to professionally crafted software built by teams of hundreds of software engineers, software designers, and user experience experts (e.g., Facebook, Gmail, Google Docs, YouTube, Evernote, Dropbox). They can read their email on all devices, download and sync photos and videos automatically, and have offline and mobile data there seamlessly when they need it. Yet research software is often built by computer science students with no experience in software engineering and human computer interaction. Overwhelmingly, users attribute their use of generic data curation software such as Microsoft Excel or Google Spreadsheets, rather than software specifically designed for language documentation, to the productivity of the user experience itself (Cathcart et al., 2012). In some cases users are so productive using Google Spreadsheets that the actual data entry of a project can be completed before an existing language documentation tool can be evaluated and/or customized (Troy and Strack, 2014).

2.2 Existing software

Fieldwork teams typically have the choice between using general-purpose content curation software (Google Spreadsheets, Evernote, Dropbox, MediaWikis, WordPress, etc.), creating/customizing their own tools, or using specialized field linguistics desktop applications such as those developed by SIL International: FieldWorks Language Explorer (FLEx),[7] Toolbox/Shoebox,[8] and/or WeSay.[9]

The SIL tools[10] require a not inconsiderable level of training in order to be used productively. However, many research teams are unable to impose lengthy training upon all team members and require tools that are easy to learn and re-learn months or years later when they return to their data. In addition, the SIL tools are tailored towards the collection of texts and the production of dictionaries and descriptive grammars based on such. However, this focus does not always accord with the needs of research-oriented fieldworkers, many of whom deal primarily in sentences elicited in isolation and grammaticality judgments.

Existing language documentation software tools, with the exception of WeSay (a collaborative dictionary tool), have only ad hoc support for collaboration (Req. 4) and inclusive language documentation (Req. 3) while the project is active, generally using a shared network drive or email with no concurrent editing. FLEx and many private tools in the language technology industry are able to support concurrent editing in most data entry situations via a Mercurial/SVN/CVS/Git repository (SIL International, 2013). However, as no permissions are built into Mercurial/SVN/CVS/Git, users with read only access must use a manual review process to offer their modifications to the project. The FLEx Send/Receive collaboration module also limits the integration of audio/video primary data; it unfortunately does not support formats used by field linguists including .ogg, .avi, .mp4, and .mov, and limits the maximum file size to 1MB (SIL International, 2013), despite the fact that most elicitation sessions or long utterances can range between 10MB and 200MB. While these scenarios may seem like rare edge cases, they can, in fact, result in teams opting not to use software designed for language documentation.

Over the past decade or so, a number of language-specific collaborative websites have arisen, examples of which are the Yurok Documentation Project (Garrett et al., 2001), the Washo

[7] http://fieldworks.sil.org/flex

[8] Toolbox is the community-supported continuation of Shoebox http://www-01.sil.org/computing/toolbox/information.htm

[9] http://www.sil.org/resources/software_fonts/wesay

[10] For reviews of FLEx and Toolbox, see Butler and van Volkinburg (2007), Rogers (2010), and Robinson et al. (2007).

Project (Yu et al., 2005; Cihlar, 2008), the Washo Mobile Lexicon (Yu et al., 2008), Karuk Dictionary and Texts (Garrett et al., 2009), and the Ilaatawaakani project (Troy and Strack, 2014). More recently, collaborative tools have arisen that, like FLEx and Toolbox, are not specific to any one language, but unlike FLEx and Toolbox, run on all devices in a web browser. In this family belong TypeCraft (Beermann and Mihaylov, 2012), the OLD (Dunham, 2014), and LingSync (Cathcart et al., 2012).

TypeCraft uses a MediaWiki UI combined with additional functionality written in Java for managing collaboration permissions and sharing. TypeCraft falls into the category of field databases designed by corpus linguists. As such it imposes upon users closed lists of categories for languages and parts of speech (Farrar, 2010), an imposition which is unacceptable to field linguists who are dealing with evolving fine-grained analyses of data categories. In addition, TypeCraft is online only, a limitation which, as Farrar (2010) correctly points out, is "not inconsiderable, especially for fieldworkers who may not have Internet access."

None of the software projects discussed in this section meet the software requirements for endangered languages fieldwork outlined in §2.1. We argue that this mismatch in requirements is nontrivial and is the reason why so much fragmentation and introduction of novel language documentation tools and software has occurred.[11]

3 New models for data collection and management

3.1 LingSync

LingSync is composed of existing and novel open source software modules (rich client-side web components and task-specific web services) which allow all stakeholders of a language documentation effort to collaboratively create corpora of primary analyzed and unanalyzed language data (Cathcart et al., 2012).

To meet the user productivity requirement (Req. 5), LingSync uses a quasi-blackboard system architecture similar to Android;[12] that is, modules can be registered to perform certain tasks, and users can discover and choose between registered modules. Similar to Praat,[13] all events in the system provide an audit trail which can be used by users,[14] but also serve as data for automated reasoning engines, should labs choose to make use of the audit data to assist in data cleaning and data quality assurance.

Based on the LingSync team's collective prior experience as field linguists, research assistants, professional lexicographers, and linguists in the language technologies industry, we hypothesize that perhaps 50% of data curation/cleaning tasks are monotonous, repetitive and consistent and thus are candidates for data manipulation best done by machines or crowdsourcing rather than by one individual human for extended periods of time. The automation of tasks in field linguistic research is rarely done, and for good reason. Unlike corpus linguistics, field linguistics seeks fine-grained analysis of novel data on under-documented languages, and data curators must be sensitive to the slightest "off" feeling of analysis which could easily be flattened by over-generalizing cleaning scripts. Automated modifications must be fully traceable so as to detect side effects of cleaning long after it has occurred. They must also be easily undoable so as not to introduce consistency or systematicity which in fact does not exist in the data.

The potential time-saving features of LingSync's system design will not bear usable data without the explicit and overarching goal of providing a user-friendly experience for both expert and novice users with differing data description vocabularies and interests (Troy and Strack, 2014). Notable user-facing features include complete UI customization, powerful searches and mapping over data sets, encryption at a field level, flexible enforcement of data consistency, social collaborative software features, an inclusive permissions system, pluggable semi-automatic glossers, numerous task-oriented web services which wrap existing libraries and scripts for audio, video, image and text analysis, two native Android GUIs

[11] We would like to point out that there are numerous other projects that have started and failed in the past 10 years which we have not had space to mention. The only stable long-term fieldwork software projects have been those which have been undertaken by the Summer Institute of Linguistics (SIL). The SIL development team is also on GitHub (https://github.com/sillsdev), a social tool for open source project management; this will likely yield technical crossover with research teams and more use of HTML5 to facilitate meeting the requirements delineated in §2.1 in future SIL software.

[12] http://developer.android.com

[13] http://praat.org

[14] In the case of Praat users are able to generate automation scripts by clicking to create a repeatable sequence of events.

which function offline (Learn X and the Elicitation Session Recorder), and five browser-based GUIs (the Prototype, Spreadsheet, Activity Feeds, Corpus Pages, Lexicon Browser), one of which functions offline and provides flexible import and export functionality. Nearly all logic is performed on the client-side which permits users to go offline and consume low bandwidth when there is limited connectivity through 3G or dial-up connections. For up-to-date examples of GUI interaction, readers are encouraged to search for LingSync on YouTube. As of April 2014 there are over 40 videos made by users demonstrating diverse features in the systems.

3.2 OLD

The OLD is software for creating web services that facilitate collaborative linguistic fieldwork. A language-specific OLD web service exposes a consistent API,[15] meaning that it can easily be used as the backend to multiple user-facing applications or as a component in a larger suite of tools. An OLD web service and the current OLD GUI together provide a number of features that respond to the requirements given in §2.1.

A language-specific OLD application allows for multiple contributors to simultaneously create, modify, browse, and search language data. This data consists of linguistic forms (i.e., morphemes, words, or phrases) that can be used to build corpora and texts. The OLD supports the integration of primary audio/video data by allowing for individual forms to be associated to any number of audio or video files (or even to subintervals of such files) and by generating representations wherein textual and audio/video data are simultaneously accessible. Data is presented in interlinear glossed text (IGT) format and individual forms, collections of forms, and texts can be exported as (Xe)LaTeX, tab-separated values (TSV), or plain text. The system provides powerful search functionality including filters over system-generated serializations of morphological analyses and, via integration with TGrep2,[16] the matching of structural patterns within treebank corpora.

Features promoting consistency include configurable orthography converters, inventory-based input validation, and the provision of visual feedback on the extent to which user-generated morphological analyses match existing lexical entries in the database. That last feature means that when a user creates a morphologically complex entry, the IGT representation indicates, via colour-coded internal links, whether the morpheme shapes and glosses match current lexical entries. It has proved to be quite useful in helping groups of fieldworkers to generate consistent morphological analyses.

3.3 LingSync/OLD

While LingSync and the OLD arose independently and consequently use different technology stacks, the teams behind the tools have largely complementary interests and are collaborating on future developments in order to combine strengths and reduce fragmentation of efforts. In the coming years, if resources permit, we hope to bring OLD's glossing UIs, logic for connecting documents to utterances as well as structural search and morphological parsing (§5.2) into the LingSync plugin architecture, with OLD UIs being used by field linguists and LingSync UIs being used by language community members and computational linguists. When referring collectively to both tools, we will henceforth use the term LingSync/OLD.

4 User adoption

In the year and a half LingSync's launch, over 300 unique users have registered; this despite the availability of a sample user (username: LingLlama, password: phoneme). We argue this demonstrates a general interest in novel, even unheard-of, language documentation software, despite the existing solutions discussed in §2.2.

Table 1 provides an overview of the corpora being edited using the system. Currently there are about 13,400 active records, 38 active users, 15 active corpora, and 1GB of primary audio/image/text data. We expect that the low ratio of active vs. registered users (12%) is due to both the multitask nature of language documentation projects and early launch of LingSync while it was still in the alpha testing and the requirements gathering phase. There are currently no published mea-

[15] The OLD API is RESTful and JavaScript Object Notation (JSON) is used as the medium of exchange throughout. This means that OLD resources (e.g., linguistic data points such as sentences) can be created, retrieved, updated, deleted, and searched using standard combinations of Hypertext Transfer Protocol (HTTP) methods and uniform resource locator (URL) patterns. The system is written in Python using the Pylons web framework (http://www.pylonsproject.org/projects/pylons-framework/about) and the relational database software MySQL.

[16] http://tedlab.mit.edu/~dr/Tgrep2/.

sures of user attrition in language documentation projects, however social websites/mobile apps developers report 30% retention rate is acceptable.[17] We will know more about rates for different stakeholders in language documentation projects as the retention rate changes over time in correlation to the release of new modules.

	Active	Investigating	In-active	Total
Public Corpora	2	1	2	5
Private Corpora	15	37	321	373
Users	38	43	220	301
Documents	13,408	2,763	4,541	23,487
Disk Size	1GB	.9GB	5.3GB	7.2GB

Table 1: Data in LingSync corpora (Feb 14, 2014). Active corpora: >300 activities; Investigating corpora: 300-10 activities; Active users: >100 activities; Investigating users: 100-10 activities.

There are currently nine language-specific OLD applications in use. In total, there are about 19,000 records (primarily sentences), 300 texts, and 20 GB of audio files. There are 180 registered users across all applications, of which 98 have entered and 87 have elicited at least one record. The applications for Blackfoot, Nata, Gitksan, Okanagan, and Tlingit are seeing the most use. The exact figures are summarized in Table 2.[18]

language	*forms*	texts	audio	GB	speakers
Blackfoot (*bla*)	8,847	171	2,057	3.8	3,350
Nata (*ntk*)	3,219	32	0	0	36,000
Gitksan (*git*)	2,174	6	36	3.5	930
Okanagan (*oka*)	1,798	39	87	0.3	770
Tlingit (*tli*)	1,521	32	107	12	630
Plains Cree (*crk*)	686	10	0	0	260
Ktunaxa (*kut*)	467	33	112	0.2	106
Coeur d'Alene (*crd*)	377	0	199	0.0	2
Kwak'wala (*kwk*)	98	1	1	0.0	585
TOTAL	19,187	324	2,599	19.8	

Table 2: Data in OLD applications (Feb 14, 2014)

The data in Table 1 and Table 2 indicate that the systems are in fact being used by language documentation teams.

5 Reusing existing tools and libraries

Both the LingSync and the OLD projects were founded with the goal of making it easier to integrate existing software libraries to better automate data curation (Req. 2) and improve data quality (Req. 4) while doing fieldwork. There have been numerous plugins in both systems to this end; however in this paper we will discuss only those which may be of most interest to computational linguists working on low-resource languages: morphological parsers in §5.1, §5.2 and §5.3 (precursors for Information Retrieval and Machine Translation tasks) and phone-level alignment of audio and text in §5.4 (a precursor for acoustic model training in Speech Recognition systems).

5.1 Existing morphological parsers

For one LingSync team working on Inuktitut, a web service was written which wraps an existing morphological analyzer for Inuktitut built in Java (Farley, 2012). This source code can be used to wrap other existing language-specific morphological analyzers.[19]

5.2 Novel morphological parsers

An OLD web service provides functionality that allows users to create any number of morphological parsers. The phonological mappings of these parsers are declared explicitly, using a formalism—context-sensitive (CS) phonological rewrite rules (Chomsky and Halle, 1968)—that is well understood by linguists. The lexicon, morphotactic rules, and parse candidate disambiguator components are automatically induced from corpora specified by the user. The fact that this implementation requires a good deal of explicit specification by the user should not be considered a demerit. By granting linguist fieldworkers control over the specification of phonological, lexical, and morphotactic generalizations, the parser functionality allows for the automatic testing of these generalizations against large data sets. This assists in the discovery of counterexamples to generalizations, thereby expediting the improvement of models and advancing linguistic research. The OLD morphological parser implementation can, of course, co-exist with and complement less

[17] There are no official published statistics; however, in answers on StackOverflow developers report averages to be 30%, cf. http://stackoverflow.com/questions/6969191/what-is-a-good-active-installs-rate-for-a-free-android-app.

[18] Note that the values in the speakers column are taken from Ethnologue (http://www.ethnologue.com) and are provided only to give a rough indication of the speaker populations of the languages. Also, the three-character codes in the first column are the ISO 639-3 (http://www-01.sil.org/iso639-3) identifiers of the languages.

[19] All modules discussed in this paper are available by searching the GitHub organization page https://github.com/opensourcefieldlinguistics

expert-dependent Machine Learning approaches to creating morphological parsers.

The core component of an OLD morphological parser is a morphophonology that is modelled as a finite-state transducer (FST)[20] and which maps transcriptions to morphological analyses, i.e., morpheme segmentations, glosses, and categories. The morphophonology FST is the composition of a phonology FST that is created explicitly by the user (using CS phonological rewrite rules) and a morphology (i.e., lexicon and morphotactic rules) that is induced from corpora constructed by the user, cf. Beesley and Karttunen (2003) and Hulden (2012). When the morphophonology returns multiple parse candidates, the system employs an N-gram language model (LM)[21] (estimated from a corpus specified by the parser's creator) to determine the most probable parse.

Preliminary tests of the OLD morphological parser implementation have been performed using data from the Blackfoot OLD[22] and the standard grammar (Frantz, 1991) and dictionary (Frantz and Russell, 1995) of the language. An initial parser implemented the phonology specified in Frantz (1991) and defined a morphology with lexical items extracted from Frantz and Russell (1995) and morphotactic rules induced from words analyzed by contributors to the system. Analysis of the performance of this parser (f-score: 0.21) confirms what researchers (Weber, 2013) have already observed, namely that the phonological and morphological generalizations of Frantz (1991) cannot account for the location of morphologically conditioned prominence (i.e., pitch accent) in Blackfoot words.

An improved Blackfoot parser, i.e., one which can predict prominence location based on the generalizations of Weber (2013), is currently under development. The phonology of this parser makes use of a novel and useful feature, viz. the ability to specify phonological transformations that are aware of categorial context. This allows the phonology to capture the distinct nominal and verbal prominence location generalizations of Blackfoot.

Since OLD morphological parsers can be created and parses retrieved entirely by issuing RESTful requests, other applications can easily make use of them. In addition, OLD morphological parser objects can be exported as .zip archives that contain all of the requisite binaries (i.e., compiled foma and MITLM files) and a Python module and executable which together allow for the parser to be used locally via the command line or from within a Python program.

5.3 Semi-supervised morphological parsers

LingSync's glosser uses a MapReduce function which efficiently indexes and transforms data to create a current "mental lexicon" of the corpus. The mental lexicon is modelled as a connected graph of morphemes, including precedence relations which are used to seed finite-state automata (Cook, 2009)[23] which represent morphological templates in the corpus. In this way the glosser is "trained" on the user's existing segmentation and glossing, and automatically "learns" as the user adds more data and the glossing/segmentation evolves over the course of data collection and analysis. LingSync has a lexicon browser component which permits users to browse the corpus via learned relations between morphemes, clean the data for consistency, enter novel data, and explicitly document generalizations on lexical nodes which might not be immediately evident in the primary data. Unlike FLEx (Black and Simons, 2006), the OLD, and WeSay, LingSync does not provide a way to explicit add rules/relations or morphemes which are not gleaned from the data. To add a morpheme or a relation users must add an example sentence to the corpus. This grounding of morphemes and rules/relations provides arguably better learning tools as collocation dictionaries and lexicon creators are always able to provide headwords and grammatical rules in context and researchers working on relations between morphemes are able to extract lists of relevant data.

5.4 Audio-transcription alignment

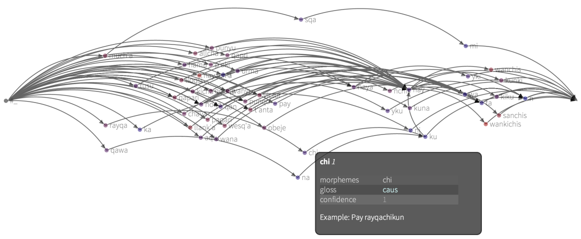
There are currently three audio web services. The first executes Sphinx speech recognition routines for languages with known language models. The second, illustrated in Figure 2a, uses

[20] FSTs are constructed using the open source finite-state compiler and C library foma: http://code.google.com/p/foma

[21] OLD N-gram LMs are estimated using MITLM: https://code.google.com/p/mitlm/.

[22] http://bla.onlinelinguisticdatabase.org/

[23] One reviewer requests more details which have not yet been published: in the interim please consult the code which is entirely open source and commented: https://github.com/OpenSourceFieldlinguistics/FieldDBGlosser



chi *1*

morphemes chi
gloss caus
confidence 1

Example: Pay rayqachikun

Figure 1: Screenshot of the Lexicon Browser, a web widget which lets users browse relations between morphemes in their corpus, clean and add declarative knowledge not found in the lexicon training process.

the Prosodylab-Aligner[24] tool (developed at the McGill Prosody Lab) to significantly automate the association of transcriptions to relevant audio clips and therefore help to provide a class of data that will prove valuable in applications such as talking dictionaries and language learning tools. The third, illustrated in Figure 2b, is a service that wraps FFmpeg[25] and Praat[26] to convert any video or audio format to .mp3 and automatically generate syllable timings and suggested utterance boundaries (De Jong and Wempe, 2009) for automatic chunking of data.

```
a) $ curl --cookie my-cookies.txt\
  --request POST\
  -F files[]=@omi_imitaa.mov\
  -F files[]=@omi_imitaa.lab\
  https://api.lingsync.org/v2/corpora/public-curldemo/
  utterances?process=align

b) $ curl --cookie my-cookies.txt\
  --request POST\
  -F files[]=@omi_imitaa.mov\
  https://api.lingsync.org/v2/corpora/public-curldemo/
  utterances?process=detect

c) $ curl --cookie my-cookies.txt\
  --request GET\
  https://api.lingsync.org/v2/corpora/public-curldemo/
  files/omi_imitaa.mp3

d) $ curl --cookie my-cookies.txt\
  --request GET\
  https://api.lingsync.org/v2/corpora/public-curldemo/
  files/omi_imitaa.TextGrid
```

Figure 2: Audio/video and text alignment via Prosodylab-Aligner web service (a), detecting utterances and syllable timing from audio/video files (b), retrieving web playable audio (c), and TextGrid results (d).

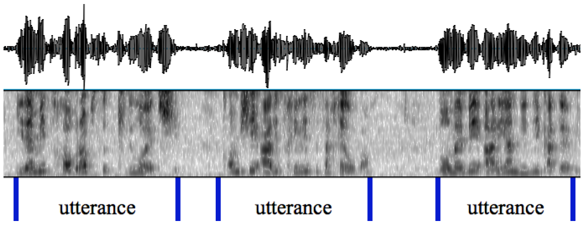


Figure 3: Screenshot of the utterance extraction process which converts any audio/video into utterance intervals encoded either as JSON or TextGrid using the PraatTextGridJS library.

6 Using LingSync/OLD

Current notable results of the LingSync/OLD project include Kartuli Glasses for Facebook (a transliterator from the Latin alphabet to the Kartuli alphabet),[27] Georgian Together for Android (a language learning app),[28] and Kartuli Speech Recognizer for Android.[29] These apps were developed in collaboration with Kartuli speakers and Kartuli software developers in Batumi, Georgia during the Spring 2014 semester.

Field linguists interested in a more detailed feature breakdown of LingSync and the OLD are encouraged to consult Cathcart et al. (2012) and Dunham (2014), respectively. Additional details on LingSync—which may be useful to those interested in developing tools with language communities or to computational linguists interested in contributing to the project—can be found in the LingSync WhitePaper (LingSync, 2012).

7 Conclusion

In this paper we hope to have illuminated some of the complexity involved in building software for endangered language documentation which has resulted in software fragmentation. We have presented LingSync/OLD, an open-ended plugin architecture which puts Software Engineering best practices and our collective experience in the language technology industry to use to address this fragmentation. The LingSync/OLD project has worked in an iterative fashion, beginning with UIs

[24] https://github.com/kylebgorman/Prosodylab-Aligner

[25] http://www.ffmpeg.org/

[26] http://www.praat.org/

[27] Chrome Store https://chrome.google.com/webstore/detail/kartuli-glasses/ccmledaklimnhjchkcgideafpglhejja

[28] Android Store https://play.google.com/store/apps/details?id=com.github.opensourcefieldlinguistics.fielddb.lessons.georgian

[29] Android Store https://play.google.com/store/apps/details?id=com.github.opensourcefieldlinguistics.fielddb.speech.kartuli

for field linguists in 2012-2013 and UIs for community members, and software libraries and training for software developers in 2013-2014. User studies and the dissemination of potentially novel language documentation and/or computational linguistics contributions are expected in 2014-2015 and in the future as the project continues to iterate. For technical updates, interested readers may view the project's completed milestones;[30] for user-facing updates, readers may visit LingSync.org and OnlineLinguisticDatabase.org.

Acknowledgements

We would like to express our deep thanks to Tobin Skinner, Elise McClay, Louisa Bielig, MaryEllen Cathcart, Theresa Deering, Yuliya Manyakina, Gretchen McCulloch, Hisako Noguchi, Brian Doherty, Gay Hazan, Oriana Kilbourn, Kim Dan Nguyen, Rakshit Majithiya, Mietta Lennes, Nivja de Jong, Ton Wempe, Kyle Gorman, Curtis Mesher, Beso Beridze, Tornike Lasuridze, Zviadi Beradze, Rezo Turmanidze, Jason Smith, Martin Gausby, Pablo Duboue, Xianli Sun, James Crippen, Patrick Littell, Michael McAuliffe as well as countless other linguistics students, computer science students and open source software developers who directly or indirectly helped build LingSync/OLD to what it is today and will be in the future. We would like to thank the ComputEL workshop reviewers, LingSync/OLD users and would-be users for providing feedback, suggestions, asking tough questions, and sending bug reports, all of which have been instrumental to the project's success and helped drive the its development. We would like to thank Faryal Abbassi, Farah Abbasi, Tamilla Paghava, Esma Chkhikvadze, Nina Gatenadze, and Mari Mgeladze for their friendship, patience and for sharing their language with us. Finally, we would like to thank Jessica Coon, Alan Bale and Michael Wagner for their guidance and challenging us to keep the user interfaces simple and yet flexible, as well as SSHRC Connection Grant (#611-2012-0001) and SSHRC Standard Research Grant (#410-2011-2401) which advocates open source approaches to knowledge mobilization and partially funded the students who have doubled as fieldwork research assistants and interns on the project. All errors and oversights are naturally our own.

[30] https://github.com/OpenSourceFieldlinguistics/FieldDB/issues/milestones?state=closed

References

Dorothee Beermann and Pavel Mihaylov. 2012. TypeCraft collaborative databasing and resource sharing for linguists. *Language Resources and Evaluation*, pages 1–23.

Kenneth R Beesley and Lauri Karttunen. 2003. Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford.*

H. Andrew Black and Gary F. Simons. 2006. The SIL FieldWorks Language Explorer approach to morphological parsing. In *Computational Linguistics for Less-studied Languages: Proceedings of Texas Linguistics Society, Austin, TX.*

Lynnika Butler and Heather van Volkinburg. 2007. Review of FieldWorks Language Explorer (FLEx). *Language Documentation & Conservation*, 1(1):100–106.

MaryEllen Cathcart, Gina Cook, Theresa Deering, Yuliya Manyakina, Gretchen McCulloch, and Hisako Noguchi. 2012. LingSync: A free tool for creating and maintaining a shared database for communities, linguists and language learners. In Robert Henderson and Pablo Pablo, editors, *Proceedings of FAMLi II: workshop on Corpus Approaches to Mayan Linguistics 2012*, pages 247–250.

N. Chomsky and M. Halle. 1968. *The Sound Pattern of English*. Harper & Row, New York.

Jonathon E. Cihlar. 2008. Database development for language documentation: A case study in the Washo language. Master's thesis, University of Chicago.

Gina Cook. 2009. Morphological parsing of Inuktitut. Ms, Concordia University, Faculty of Engineering and Computer Science.

David Costa. 2012. Surveying the sources on the Myaamia language. In *Proceedings of the 2012 Myaamiaki Conference*.

N.H. De Jong and T Wempe. 2009. Praat script to detect syllable nuclei and measure speech rate automatically. *Behavior research methods*, 41(2):385–390.

Joel Dunham. 2014. Online Linguistic Database documentation. http://online-linguistic-database.readthedocs.org, March.

Benoit Farley. 2012. The Uqailaut project. http://www.inuktitutcomputing.ca, January.

Scott Farrar. 2010. Review of TypeCraft. *Language Documentation & Conservation*, 4:60–65.

Donald G. Frantz and Norma Jean Russell. 1995. *Blackfoot Dictionary of Stems, Roots, and Affixes*. Toronto: University of Toronto Press.

Donald G. Frantz. 1991. *Blackfoot Grammar*. Toronto: University of Toronto Press.

Andrew Garrett, Juliette Blevins, Lisa Conathan, Anna Jurgensen, Herman Leung, Adrienne Mamin, Rachel Maxson, Yoram Meroz, Mary Paster, Alysoun Quinby, William Richard, Ruth Rouvier, Kevin Ryan, and Tess Woo. 2001. The Yurok language project. http://linguistics.berkeley.edu/~yurok/index.php, January.

Andrew Garrett, Susan Gehr, Line Mikkelsen, Nicholas Baier, Kayla Carpenter, Erin Donnelly, Matthew Faytak, Kelsey Neely, Melanie Redeye, Clare Sandy, Tammy Stark, Shane Bilowitz, Anna Currey, Kouros Falati, Nina Gliozzo, Morgan Jacobs, Erik Maier, Karie Moorman, Olga Pipko, Jeff Spingeld, and Whitney White. 2009. Karuk dictionary and texts. http://linguistics.berkeley.edu/~karuk/links.php, January.

Jeff Good. 2012a. 'Community' collaboration in Africa: Experiences from northwest Cameroon. *Language Documentation and Description*, 11(1):28–58.

Jeff Good. 2012b. Valuing technology: Finding the linguist's place in a new technological universe. In Louanna Furbee and Lenore Grenoble, editors, *Language documentation: Practice and values*, pages 111–131. Benjamins, Amsterdam.

K David Harrison. 2007. *When Languages Die: The Extinction of the World's Languages and the Erosion of Human Knowledge*. Oxford University Press.

M. Hulden. 2012. foma: finite state compiler and C library (documentation). https://code.google.com/p/foma/w/list.

George Ironstrack. 2012. Miloniteeheetaawi eehinki pimihkanaweeyankwi: Let's reflect on how far we have traveled. In *Proceedings of the 2012 Myaamiaki Conference*.

Wesley Leonard. 2012. Your language isn't extinct: the role of Myaamia in Language Reclamation. In *Proceedings of the 2012 Myaamiaki Conference*.

LingSync. 2012. WhitePaper. http://OpenSourceFieldlinguistics.github.io/FieldDB/, January.

Stuart Robinson, Greg Aumann, and Steven Bird. 2007. Managing fieldwork data with ToolBox and the Natural Language Toolkit. *Language Documentation & Conservation*, 1(1):44–57.

Chris Rogers. 2010. Review of FieldWorks Language Explorer (FLEx) 3.0. *Language Documentationation & Conservation*, 4:78–84.

R. Schroeter and N. Thieberger. 2006. EOPAS, the EthnoER online representation of interlinear text. In Sebastian Nordoff, editor, *Sustainable Data from Digital Fieldwork*. University of Sydney, Sydney.

SIL International. 2013. *Technical Notes on FieldWorks Send/Receive*. http://fieldworks.sil.org/wp-content/TechnicalDocs/, November.

Nick Thieberger. 2012. Using language documentation data in a broader context. In Frank Seifart, Geoffrey Haig, Nikolaus P. Himmelmann, Dagmar Jung, Anna Margetts, and Paul Trilsbeek, editors, *Potentials of Language Documentation: Methods, Analyses, and Utilization*. University of Hawai'i Press, Honolulu.

Doug Troy and Andrew J. Strack. 2014. Metimankwiki kimehšoominaanaki - we follow our ancestors trail: Sharing historical Myaamia language documents across myaamionki. In *Proceedings of the 2014 Myaamiaki Conference*.

N. Weber. 2013. Accent and prosody in Blackfoot verbs. http://www.academia.edu/4250143/Accent_and_prosody_in_Blackfoot_verbs.

Alan Yu, Ryan Bochnak, Katie Franich, Özge Sarigul, Peter Snyder, Christina Weaver, Juan Bueno-Holle, Matt Faytak, Eric Morley, and Alice Rhomieux. 2005. The Washo project. http://washo.uchicago.edu/dictionary/dictionary.php, January.

Alan Yu, Ryan Bochnak, Katie Franich, Özge Sarigul, Peter Snyder, Christina Weaver, Juan Bueno-Holle, Matt Faytak, Eric Morley, and Alice Rhomieux. 2008. The Washo mobile lexicon. http://washo.uchicago.edu/mobile/, January.