Towards a Data Model for the Universal Corpus

Steven Abney
University of Michigan
abney@umich.edu

Steven Bird
University of Melbourne and
University of Pennsylvania
sbird@unimelb.edu.au

Abstract

We describe the design of a comparable corpus that spans all of the world's languages and facilitates large-scale cross-linguistic processing. This Universal Corpus consists of text collections aligned at the document and sentence level, multilingual wordlists, and a small set of morphological, lexical, and syntactic annotations. The design encompasses submission, storage, and access. Submission preserves the integrity of the work, allows asynchronous updates, and facilitates scholarly citation. Storage employs a cloud-hosted filestore containing normalized source data together with a database of texts and annotations. Access is permitted to the filestore, the database, and an application programming interface. All aspects of the Universal Corpus are open, and we invite community participation in its design and implementation, and in supplying and using its data.

1 Introduction

We have previously proposed a community dataset of annotated text spanning a very large number of languages, with consistent annotation and format that enables automatic cross-linguistic processing on an unprecedented scale (Abney and Bird, 2010). Here we set out the data model in detail, and invite members of the computational linguistics community to begin work on the first version of the dataset.

The targeted annotation generalizes over three widely-used kinds of data: (1) simple bitexts, that is, tokenized texts and their translations, which are widely used for training machine translation systems; (2) interlinear glossed text (IGT), which adds lemmas, morphological features and parts of speech, and is the de facto standard in the documentary linguistics literature; and (3) dependency parses, which add a head pointer and relation name for each word, and are gaining popularity as representations of syntactic structure. We do not expect all texts to have equal richness of annotation; rather, these are the degrees of annotation we wish to explicitly accommodate. Keeping the annotation lightweight is a primary desideratum.

We strive for inclusion of as many languages as possible. We are especially interested in languages outside of the group of 30 or so for which there already exist non-trivial electronic resources. Optimistically, we aim for a *universal* corpus, in the sense of one that covers a widely representative set of the world's languages and supports inquiry into universal linguistics and development of language technologies with universal applicability.

We emphasize, however, that even if completely successful, it will be *a* universal corpus and not *the* universal corpus. The term "universal" should emphatically not be understood in the sense of encompassing all language annotation efforts. We are not proposing a standard or a philosophy of language documentation, but rather a design for one particular resource. Though the goals with regard to language coverage are unusually ambitious, for the sake of achievability we keep the targeted annotation as simple as possible. The result is intended to be a single, coherent dataset that is very *broad* in language coverage, but very *thin* in complexity of annotation.

Proceedings of the 4th Workshop on Building and Using Comparable Corpora, pages 120–127,
49th Annual Meeting of the Association for Computational Linguistics,
Portland, Oregon, 24 June 2011. ©*2011 Association for Computational Linguistics*

Finally, the development of the corpus is an unfunded, all-volunteer effort. It will only come about if it wins community buy-in, in the spirit of collaborative efforts like Project Gutenberg. We formulate it as a cooperation among data providers and hosting services to provide data in a manner that creates a single, seamless dataset from the user perspective. This paper is a first draft of a "cooperative agreement" that could achieve that goal.

2 A lightweight model for multilingual text

2.1 Media and annotation

In documentary linguistics, a distinction is made between language documentation, whose concern is the collection of primary documentation such as speech recordings and indigenous written works, and language description, whose concern is the annotation and organization of the primary material (Himmelmann, 1998). We make a similar distinction between media files and annotation, where "annotation" is understood broadly to include all processing steps that make the linguistic contents more explicit, including plain text rendering, sentence segmentation, and alignment of translations.

The Corpus consists of annotated documents, in the sense of primary documents with accompanying annotation. There are many efforts at collecting documentation for a broad range of languages; what makes this Corpus distinct is its focus on annotation. Accordingly, we assume that media files and annotation are handled separately.

For media, the Language Commons collection in the Internet Archive is a recently-established repository for redistributable language data that we view as the primary host.[1] For the annotation database, a primary data host remains to be established, but we have identified some options. For example, Amazon Web Services and the Talis Connected Commons have free hosting services for public data sets.

2.2 The data model in brief

In order to keep the barriers to participation as low as possible, we have made our target for annotation as simple as possible. The data model is summarized in Figure 1. We distinguish between **aligned texts** (or parallel texts) and **analyzed texts** (comparable texts).

Semantically, the entire collection of aligned texts constitutes a matrix whose columns are languages and whose rows are texts. We limit attention to three levels of granularity: document, sentence, and word. Each cell is occupied by a string, the typical length of the string varying with the granularity. We expect the matrix to be quite sparse: most cells are empty.

The collection of analyzed texts consists, semantically, of one table per language. The rows represent words and the columns are properties of the words. The words may either be tokens in a sentence analysis, as suggested by the examples, or types representing dictionary information. The tables are comparable, in the sense that they have a common format and are conducive to language-independent processing, but they are not parallel: the i-th word in the German table has nothing to do with the i-th word in the Spanish table.

The tables in Figure 1 constitute the bulk of the data model. In addition, we assume some auxiliary information (not depicted) that is primarily organizational. It includes an association between documents and sentences, the location of documents and sentences within media files (if applicable), a grouping of table rows into "files," and a grouping of files into "works." Metadata such as revision information is attached to files and works. We return below to the characterization of this auxiliary information.

In contrast to current standard practice, we wish to emphasize the status of aligned and analyzed text as annotation of primary documents represented by media files such as speech recordings or page images, and we wish to maintain explicit connections between annotations and primary documents. We do not insist that the underlying media files be available in all cases, but we hope to identify them when possible. However, we focus on storage of the annotation; we assume that media files are in a separate store, and referenced by external URIs.

2.3 Two implementations: filestore and database

The data model is abstract, and is implemented in a couple of ways for different purposes. For distribution on physical medium or by download, it is most convenient to implement the data model as actual

[1] `http://www.archive.org/details/LanguageCommons`

Aligned Texts

	deu	spa	fra	eng	...
d_1	*sie..*	*ella..*	*elle..*	*she..*	
d_2					
⋮					
s_1					
s_2					
⋮					
w_1					
w_2					
⋮					

Analyzed Texts

deu

	sent	form	lemma	morph	pos	gloss	head	rel
w_1	s_1	*Kühe*	*Kuh*	PL	*N*	*cow*	2	SBJ
w_2	s_1	*sind*	*sein*	PL	*V*	*be*	0	ROOT
⋮								

spa

	sent	form	lemma	morph	pos	gloss	head	rel
w_1	s_2	*estas*	*este*	F.PL	*D*	*this*	2	SPC
w_2	s_2	*floras*	*flora*	F.PL	*N*	*flower*	3	SBJ
⋮								

⋮

Figure 1: An overview of the targeted annotation: Aligned Texts in a single matrix having three levels of granularity (document, sentence, word), and Analyzed Texts grouped by language and annotated down to the word level with morphological, lexical and syntactic information.

files. Each file contains information corresponding to some slice of a table, and the structure of the table is encoded in the file format. On the other hand, web services are often implemented as databases, making an implementation of the abstract model as a database desirable.

A file-based implementation is most familiar, and most existing resources are available as file collections. However, even when different existing resources have similar semantics, such as different parallel text collections, there is considerable variety in the organization and representation of the information. In order to work with multiple such sources, a substantial amount of housekeeping is required. One can view our proposed filestore as a normalized form that removes the diversity that only gets in the way of efficient cross-language processing. Indeed, our proposed format for analyzed text hews intentionally close to the format used in the CoNLL dependency-parsing shared tasks, which provided a normal form into which data from multiple treebanks was mapped (Buchholz et al., 2006).

When an existing resource is included in the Corpus, we assume that it remains externally available in its original form, but a copy is imported into the Corpus filestore in which every file has been preprocessed into one of a set of simple file formats implementing the model of Figure 1, following a consistent scheme for filenames, with utf8 character encoding, and capturing any available alignment information in an auxiliary table. Distribution of the Corpus via physical media or download simply involves copying the filestore.

The filestore is organized around material provided by individual data providers, or "authors," and maintains the identity of a data provider's contribution as a distinct intellectual "work." Works provide an appropriate unit to which to attach edition and rights metadata.

In addition to the filestore, the texts and alignments are imported into a collection of database tables that can be queried efficiently.

In section 3 we describe a simple file-based implementation of the data model, and show the variety of familiar file types that find a natural place in the model. In section 4 we describe the tabular storage model.

3 Filestore implementation

Despite the simplicity of the data model, it captures a substantial, even surprising, variety of commonly-used textual data file types.

Document-aligned text. Parallel corpora are most commonly aligned at the document level. Typically, each translation of a document is contained in a file, and there is some way of indicating which files are mutual translations of the same document. The contents of a file, as a single string, represents one cell in the Aligned Text matrix in Figure 1 (at the "document" level of granularity). A document, comprising a collection of mutual translations, corresponds to a row of the matrix.

As normal form, we propose the convention of using filenames that incorporate a language identifier and a document identifier. For example, `1001-eng.txt` and `1001-deu.txt` are the English and German files representing mutual translations of some hypothetical document 1001.

Language identifiers are ISO 639-3 language code, supplemented by the Linguist List local-use codes and subgroup and dialect identifiers.

Sentence-aligned text. At a finer grain, parallel corpora may be aligned at the sentence level. Each file contains the translation of one document, segmented into one sentence per line. Our normal form uses the same filename convention as for document-aligned text, to indicate which files are mutual translations. We use the file suffix "`.snt`" to indicate a file with one sentence per line. This incidentally indicates which document a set of sentences came from, since the filenames share a document identifier. For example, the file `1001-deu.snt` contains the sentence-segmented version of `1001-deu.txt`.

In the canonical case, each file in a group of aligned files contains the same number of sentences, and the sentences line up one-to-one. The group of aligned files corresponds to a set of rows in the Aligned Text matrix, at the "sentence" level of granularity.

There are cases in which the sentence alignment between documents is not one-to-one. Even in this case, we can view the alignment as consisting of a sequence of "beads" that sometimes contain multiple sentences in one language. If we normalize the file to one in which the group of sentences belonging to a single bead are concatenated together as a "translational unit," we reduce this case to the one-to-one case, though we do lose the information about orthographic sentence boundaries internal to a bead.

Preserving the original sentences would necessitate an extension to the data model. A typical approach is to store the alignments in a table, where n-way alignments are indicated using n-tuples of integers. We leave this as a point for future consideration. We also put aside consideration of word-level document alignment.

Translation dictionaries. A translation dictionary contains word translations in multiple languages. One representation looks just like sentence-aligned text, except that each file contains one entry per line instead of one sentence per line. Each file in an aligned set contains the same number of entries, and the entries line up one-to-one across files. This is the representation we take as our normal form. We also use the same filename convention, but with suffix `.tdi` for translation dictionary.

A translation dictionary corresponds to a set of rows in the Aligned Text matrix, at the "word" level of granularity. A translation dictionary would typically be derived from a large number of text documents, so each translation dictionary will typically have a unique document identifier, and will not align with files at the sentence or document granularity.

Transcriptions and segmentations. When one begins with a sound recording or with page images from a print volume that has been scanned, a first step is conversion to plain text. We will call this a "transcription" both for the case where the original was a sound file and for the case where the original was a page image. Transcriptions fit into our data model as the special case of "document-aligned text" in which only one language is involved. We assume that the Aligned Text matrix is sparse, and this is the extreme case in which only one cell in a row is occupied. The connection between the transcript's document identifier and the original media file is recorded in an auxiliary metadata file.

After transcription, the next step in processing is to identify the parts of the text that are natural language (as opposed to markup or tables or the like), and to segment the natural language portion into sentences. The result is sentence-segmented text. Again, we treat this as the special case of sentence-aligned text in which only one language is involved.

Analyzed text. A variety of different text file types can be grouped together under the heading of analyzed text. The richest example we consider is dependency parse structure. One widely-used file representation has one word token per line. Each line consists of tab-separated fields containing attributes of the word token. There is some variation in the attributes that are specified, but the ones used in the Analyzed Text tables of our data model are typical, namely: sentence identifier, wordform, lemma, morphological form, gloss, part of speech, head (also called *governor*), and relation (also called *role*). Sentence boundaries are not represented as tokens; rather, tokens belonging to the same sentence share the same value for sentence identifier. We continue with the same filename convention as before; for Analyzed Text files, the suffix is `.tab`.

Many different linguistic annotations are naturally represented as special cases of Analyzed Text.

- Tokenized text in "vertical format" is the special case in which the only column is the wordform column. We include the sentence ID column as well, in lieu of sentence-boundary tokens.
- POS-tagged text adds the part of speech column.
- The information in the word-by-word part of interlinear glossed text (IGT) typically includes the wordform, lemma, morph, and gloss; again we also include the sentence ID column.
- A dependency parse, as already indicated, is the case in which all columns are present.

In addition, the format accommodates a variety of monolingual and multilingual lexical resources. Such lexical resources are essential, whether manually curated or automatically extracted.

- A basic dictionary consists of a sequence of entries, each of which contains a lemma, part of speech, and gloss. Hence a dictionary is naturally represented as analyzed text containing just those three columns. The entries in a dictionary are word types rather than word tokens, so the wordform and sentence ID columns are absent.
- If two or more lexicons use the same glosses, the lexicons are implicitly aligned by virtue of the glosses and there is no need for overt alignment information. This is a more flexible representation than a translation dictionary: unlike a translation dictionary, it permits multiple words to have the same gloss (synonyms), and it adds parts of speech.

4 Database implementation

An alternative implementation, appropriate for deployment of the Corpus as a web service, is as a normalized, multi-table database. In this section we drill down and consider the kinds of tables and records that would be required in order to represent our abstract data model. We will proceed by way of example, for each of the kinds of data we would like to accommodate. Each example is displayed as a record consisting of a series of named fields.

Note that we make no firm commitment as to the physical format of these records. They could be serialized as XML when the database is implemented as a web service. Equally, they could be represented using dictionaries or tuples when the database is accessed via an application program interface (API). We will return to this later.

4.1 The Aligned Text matrix

The Aligned Text matrix is extremely sparse. We use the more flexible representation in which each matrix cell is stored using a separate record, where the record specifies (index, column) pairs. For example, the matrix row

	deu	spa	fra
d_1	*Sie...*	*Ella...*	
d_2	*Mein...*		*Mon...*

is represented as

Document Table

DID	LANG	TEXT
1	*deu*	*Sie...*
1	*spa*	*Ella...*
2	*deu*	*Mein...*
2	*fra*	*Mon...*

(The ellipses are intended to indicate that each cell contains the entire text of a document.) We have also added an explicit document ID.

When we consider entries at the sentence and word levels, we require both a document ID and sentence or word IDs within the document. Figure 2 shows an example of two sentences from the same document, translated into two languages. Note that we can think of DID + LANG as an identifier for a monolingual document instance, and DID + LANG + SID identifies a particular sentence in a monolingual document.

DID	LANG	SID	TEXT
1	deu	1	Der Hund bellte.
1	eng	1	the dog barked.
1	deu	2	Mein Vater ist Augenarzt.
1	eng	2	My father is an optometrist.

Figure 2: Two sentences with two translations. These are sentence table records.

In short, we implement the Aligned Text matrix as three database tables. All three tables have columns DID, LANG, and TEXT. The sentence table adds SID, and the word table adds WID instead of SID. (The words are types, not tokens, hence are not associated with any particular sentence.)

4.2 The Analyzed Text tables

The implementation of the Analyzed Text tables is straightforward. We add a column for the document ID, and we assume that sentence ID is relative to the document. We also represent the word token ID explicitly, and take it to be relative to the sentence. Finally, we add a column for LANG, so that we have a single table rather than one per language.

The first record from the German table in Figure 1 is implemented as in Figure 3. This is a record from a dependency parse. Other varieties of analyzed text leave some of the columns empty, as discussed in the previous section.

There is a subtlety to note. In the sentence table, the entry with DID 1, SID 1, and LANG “deu” is understood to be a translation of the entry with DID 1, SID 1, and LANG “eng.” That is not the case with records in the analyzed-text table. Word 1 in the English sentence 1 of document 1 is not necessarily a translation of word 1 in the German sentence 1 of document 1.

A few comments are in order about the meanings of the columns. The wordform is the attested, inflected form of the word token. The LEMMA provides the lexical form, which is the headword under which one would find the word in a dictionary. The MORPH field provides a symbolic indicator of the relationship between the lemma and the wordform. For example, “Kühe” is the PL form of the lemma “Kuh.”

This approach encompasses arbitrary morphological processes. For example, Hebrew *lomedet* may be represented as the PRESPTC.FEM.SG form of *lmd,* (“to learn”).

When we represent dictionaries, the records are word types rather than word tokens. We assign a document ID to the dictionary as a whole, but by convention take the SID to be uniformly 0.

Ultimately, the POS and GLOSS fields are intended to contain symbols from controlled vocabularies. For the present, the choice of controlled vocabulary is up to the annotator. For the GLOSS field, an option that has the benefit of simplicity is to use the corresponding word from a reference language, but one might equally well use synset identifiers from WordNet, or concepts in some ontology.

4.3 The auxiliary tables

The auxiliary tables were not shown in the abstract data model as depicted in Figure 1. They primarily include metadata. We assume a table that associates each document ID with a work, and a table that provides metadata for each work. The Corpus as a whole is the sum of the works.

In the spirit of not duplicating existing efforts, we “outsource” the bulk of the metadata to OLAC (Simons and Bird, 2003). If a work has an OLAC entry, we only need to associate the internal document ID to the OLAC identifier.

There is some metadata information that we would like to include for which we cannot refer to OLAC.

- Provenance: how the annotation was constructed, e.g., who the annotator was, or what software was used if it was automatically created.
- Rights: copyright holder, license category chosen from a small set of interoperable licenses.
- Standards: allows the annotator to indicate which code sets are used for the MORPH, POS, and GLOSS fields. We would like to be able to specify a standard code set for each, in the same way that we have specified ISO 639-3 for language codes. Consensus has not yet crystallized around any one standard, however.

The auxiliary tables also associate documents with media files. We assume a table associating document IDs with a media files, represented by

DID	LANG	SID	WID	FORM	LEMMA	MORPH	POS	GLOSS	HEAD	REL
123	deu	1	1	Kühe	Kuh	PL	N	cow	2	SBJ

Figure 3: A single word from a dependency parse. This is a record from the analyzed-text table.

their URLs, and a table associating sentences (`DID` + `SID`) with locations in media files.

Note that, as we have defined the file and tabular implementations, there is no need for an explicit mapping between document IDs and filenames. A filename is always of the form *did*–*lang*.*suffix,* where the suffix is `.txt` for the document table, `.snt` for the sentence table, `.tdi` for the word table, and `.tab` for the analyzed-text table. Each file corresponds to a set of records in one of the tables.

5 Cloud Storage and Interface

A third interface to the Corpus is via an application programming interface. We illustrate a possible Python API using Amazon SimpleDB, a cloud-hosted tuple store accessed via a web services interface.[2] An "item" is a collection of attribute-value pairs, and is stored in a "domain." Items, attributes, and domains are roughly equivalent to records, fields, and tables in a relational database. Unlike relational databases, new attributes and domains can be added at any time.

Boto is a Python interface to Amazon Web Services that includes support for SimpleDB.[3] The following code shows an interactive session in which a connection is established and a domain is created:

```
>>> import boto
>>> sdb = boto.connect_sdb(PUBLIC_KEY, PRIVATE_KEY)
>>> domain = sdb.create_domain(’analyzed_text’)
```

We can create a new item, then use Python's dictionary syntax to create attribute-value pairs, before saving it:

```
>>> item = domain.new_item(’123’)
>>> item[’DID’] = ’123’
>>> item[’LANG’] = ’deu’
>>> item[’FORM’] = ’Kühe’
>>> item[’GLOSS’] = ’cow’
>>> item[’HEAD’] = ’2’
>>> item.save()
```

Finally, we can retrieve an item by name, or submit a query using SQL-like syntax.

```
>>> sdb.get_attributes(domain, ’123’)
’LANG’: ’deu’, ’HEAD’: ’2’, ’DID’: ’123’,
’FORM’: ’Kühe’, ’GLOSS’: ’cow’
>>> sdb.select(domain,
... ’select DID, FORM from analyzed_text
... where LANG = "deu"’)
[’DID’: ’123’, ’FORM’: ’Kühe’]
```

We have developed an NLTK "corpus reader" which understands the Giza and NAACL03 formats for bilingual texts, and creates a series of records for insertion into SimpleDB using the Boto interface. Other formats will be added over time.

Beyond the loading of corpora, a range of query and report generation functions are needed, as illustrated in the following (non-exhaustive) list:

- `lookup(lang=ENG, rev="1.2b3", ...)`: find all items which have the specified attribute values, returning a list of dictionaries; following Python syntax, we indicate this variable number of keyword arguments with `**kwargs`.
- `extract(type=SENT, lang=[ENG, FRA, DEU], **kwargs)`: extract all aligned sentences involving English, French, and German, which meet any further constraints specified in the keyword arguments. (When called `extract(type=SENT)` this will extract all sentence alignments across all 7,000 languages, cf Figure 1.)
- `dump(type=SENT, format="giza", lang=[ENG, FRA], **kwargs)`: dump English-French bitext in Giza format.
- `extract(type=LEX, lang=[ENG, FRA, ...], **kwargs)`: produce a comparative wordlist for the specified languages.
- `dump(type=LEX, format="csv", lang=[ENG, FRA, ...], **kwargs)`: produce the wordlist in comma-separated values format.

Additional functions will be required for discovery (which annotations exist for an item?), navigation (which file does an item come from?), citation (which publications should be cited in connection with these items?), and report generation (what type and quantity of material exists for each language?).

[2] `http://aws.amazon.com/simpledb/`

[3] `http://code.google.com/p/boto/`

Further functionality could support annotation. We do not wish to enable direct modification of database fields, since everything in the Corpus comes from contributed corpora. Instead, we could foster user input and encourage crowdsourcing of annotations by developing software clients that access the Corpus using methods such as the ones already described, and which save any new annotations as just another work to be added to the Corpus.

6 Further design considerations

Versioning. When a work is contributed, it comes with (or is assigned) a version, or "edition." Multiple editions of a work may coexist in the Corpus, and each edition will have distinct filenames and identifiers to avoid risk of collision. Now, it may happen that works reference each other, as when a base text from one work is POS-tagged in another. For this reason, we treat editions as immutable. Modifications to a work are accumulated and released as a new edition. When a new edition of a base text is released, stand-off annotations of that text (such as the POS-tagging in our example) will need to be updated in turn, a task that should be largely automated. A new edition of the annotation, anchored to the new edition of the base text, is then released. The old editions remain unchanged, though they may be flagged as obsolete and may eventually be deleted.

Licensing. Many corpora come with license conditions that prevent them from being included. In some cases, this is due to license fees that are paid by institutional subscription. Here, we need to explore a new subscription model based on access. In some cases, corpus redistribution is not permitted, simply in order to ensure that all downloads occur from one site (and can be counted as evidence of impact), and so that users agree to cite the scholarly publication about the corpus. Here we can offer data providers a credible alternative: anonymized usage tracking, and an automatic way for authors to identify the publications associated with any slice of the Corpus, facilitating comprehensive citation.

Publication. The Corpus will be an online publication, with downloadable dated snapshots, evolving continually as new works and editions are added. An editorial process will be required, to ensure that contributions are appropriate, and to avoid spamming. A separate staging area would facilitate checking of incoming materials prior to release.

7 Conclusion

We have described the design and implementation of a Universal Corpus containing aligned and annotated text collections for the world's languages. We follow the same principles we set out earlier (Abney and Bird, 2010, 2.2), promoting a community-level effort to collect bilingual texts and lexicons for as many languages as possible, in a consistent format that facilitates machine processing across languages. We have proposed a normalized filestore model that integrates with current practice on the supply side, where corpora are freestanding works in a variety of formats and multiple editions. We have also devised a normalized database model which encompasses the desired range of linguistic objects, alignments, and annotations. Finally, we have argued that this model scales, and enables a view of the Universal Corpus as a vast matrix of aligned and analyzed texts spanning the world's languages, a radical departure from existing resource creation efforts in language documentation and machine translation.

We invite participation by the community in elaborating the design, implementing the storage model, and populating it with data. Furthermore, we seek collaboration in using such data as the basis for large-scale cross-linguistic analysis and modeling, and in facilitating the creation of easily accessible language resources for the world's languages.

References

Steven Abney and Steven Bird. 2010. The Human Language Project: building a universal corpus of the world's languages. In *Proc. 48th ACL*, pages 88–97. Association for Computational Linguistics.

Sabine Buchholz, Erwin Marsi, Yuval Krymolowski, and Amit Dubey. 2006. CoNLL-X shared task: Multilingual dependency parsing. `http://ilk.uvt.nl/conll/`. Accessed May 2011.

Nikolaus P. Himmelmann. 1998. Documentary and descriptive linguistics. *Linguistics*, 36:161–195.

Gary Simons and Steven Bird. 2003. The Open Language Archives Community: An infrastructure for distributed archiving of language resources. *Literary and Linguistic Computing*, 18:117–128.