Annotation representation and File conversion tool

Brigitte Bigi

ANNOTATION REPRESENTATION AND FILE CONVERSION TOOL

BRIGITTE BIGI
Laboratoire Parole et Langage, CNRS, Aix Marseille Univ.
5 avenue Pasteur, 13100 Aix-en-Provence, FRANCE

ABSTRACT
Annotating corpora is of crucial importance in Corpus Linguistics. Linguistics annotation, especially when dealing with multiple domains, makes use of different tools within a given project. More and more annotated corpora are now available, and so are tools to annotate automatically and/or manually. Due to the diversity of linguistic phenomena, annotation tools lead to a variety of models, theories and formalisms. This diversity results in heterogeneous description formats, each tool developing its own framework. Then, none of the annotation tools are directly interoperable, each one using a native format, some of them on top of XML, some others developing an ad hoc markup language. The mapping between user formats is then an important issue. This paper presents an efficient annotation representation framework and the related tool to convert from/to some of the existing annotation file formats of various annotation software tools for audio recordings.

1 Introduction

Corpus Linguistics is a Computational Linguistics sub-field which aims to study the language as expressed in corpora. Nowadays, *Annotation, Abstraction and Analysis* - the 3A from (Wallis and Nelson 2001), is a common perspective in this field. *Annotation* consists of the application of a scheme to recordings (text, audio, video...). *Abstraction* consists of the mapping of terms in the scheme to terms in a theoretically motivated model or dataset. *Analysis* consists of statistically probing, manipulating and generalizing from the dataset. Within these definitions, this paper is part of *Annotation* field, focusing on linguistic annotations of audio recordings.

Annotating speech recordings is relevant for many sub-fields of linguistics such as phonetics, prosody or discourse studies. Corpora are annotated thanks to the use of specialized software. Generally, “different annotation tools are designed and used to annotate the audio and video contents of a corpus that can later be merged in query systems or databases” (Abuczki and Baiat Ghazaleh 2013). To be useful for purposes such as qualitative or quantitative analyses, the annotations of recordings must be time-synchronized (i.e. time-aligned). Temporal information makes it possible to describe behavior or actions of different subjects that happen at the same time. Indeed, “when multiple annotations are integrated into a single data set, inter-relationships between the annotations can be explored both qualitatively (by using database queries that combine levels) and quantitatively (by running statistical analyses or machine learning algorithms)” (Chiarcos 2008).

For a researcher looking for an annotation software tool, it might be difficult to select the most appropriate one. See Rohlfing et al. (2006) for a comparison of multimodal annotation tools. To decide about usefulness and usability of a software, it is advisable to consider the software license, the portability, the ease of use, the strengths/weaknesses for specific annotation purposes, the type of data or analysis the tool/software is specifically designed to complete and the software compatibility with other annotated data, i.e. the availability of files to be imported from and exported to several other data formats.

The choice of a software is then of crucial importance: it determines the framework that will be utilized during the annotation process and the suitability of the annotated files. Due to the diversity of linguistic phenomena, annotation tools lead to a variety of models, theories and formalisms. This diversity results in heterogeneous description formats, each tool developing its own framework. As a consequence, none of the annotation tools are directly interoperable, each one using a native format: some of them on top of XML, some others developing an *ad hoc* markup language.

SPPAS – the automatic annotation and analysis of speech, is a recent computer software tool designed and developed by the author to handle multiple language corpora and/or tasks in the same software environment (Bigi 2015). SPPAS allows to automatically create and to analyze annotations; it offers a *fully-automatic or semi-automatic annotation process*, including for speech segmentation (Bigi 2011, Bigi 2012). SPPAS includes a generic enough annotation representation framework that allows it to import from and export to a variety of file formats, which is the purpose of this paper. It also proposes its own native XML-based format.

After a background on data representation, this paper briefly presents the annotation representation framework of SPPAS and the related tool to convert from/to some of the existing file formats of various software. Such tool is illustrated via several conversions of one file of the set of data distributed for the International Workshop “Speech audio archives: preservation, restoration, annotation, aimed at supporting the linguistic analysis”, organized by Accademia Nazionale dei Lincei, in Roma Italy in May, 2017.

2 Background on data representation

2.1 Standardization of annotation representation frameworks

Nowadays, most of linguistic annotation projects require a rich framework in order to deal with all the information levels involved. For example, dialog act annotation involves multi-dimensional communicative functions that are organized in a complex taxonomy. Manipulating linguistic annotated resources requires to be independent from the coding format which implies two kinds of pre-requisites:

1. to specify and organize the information to be encoded independently from the constraints or restriction of the format (or the annotation tool);
2. to encode the information into an exchange format, readable whatever the edition or annotation system.

One way toward interoperability is the standardization of annotation frameworks. "The ISO/TEI specifications implement the full power of feature structures and define inheritance, unification, and subsumption mechanisms over the structures, thus enabling the representation of linguistic information at any level of complexity." (Ide 2007). Recently, Liégeois et al. (2015) proposed to extend the TEI and to use it as a pivot format for oral and multimodal language corpora. However, this initiative is in its early stage and despite standardization efforts it is unlikely that a unique generic schema shared by all annotation tools emerge.

2.2 Annotation tool frameworks

Trees or graphs are a very common formalism used for various linguistic representation levels in annotation tools. Syntactic trees are the clearest example of such linguistic representations. Unlike other linguistic annotations data structures of speech/video annotated data are "tiers", "layers" or "tracks" of annotations. In that context, a *Tier* consists of series of *Annotation* instances, each one defined by a temporal localization (mostly an interval) and a label. It results in the following advantages:

- inclusion of various annotation types in the same document;
- flexible merging of documents;
- easy comparison of annotations (mainly on the basis of temporal information);
- representation of alternative/concurrent annotations, by means of separated tiers;
- representation of partial annotations (tiers with "holes");
- such annotation representation framework is not theory-dependent.

Fortunately, as shown in the left-vs-right illustrations of Figure 1, some mapping is possible between representations. In this example, we limited the annotations to: syntactic groups (g), syntax categories (c), words (w), lemmas (l), phonemes (p), syllables (s), left hand movements (lh), right hand movements (lr). Tiers can be mapped into graphs and vice-versa if graphs have

"categorized-nodes" like it is represented by various intensities of gray in Figure 1.

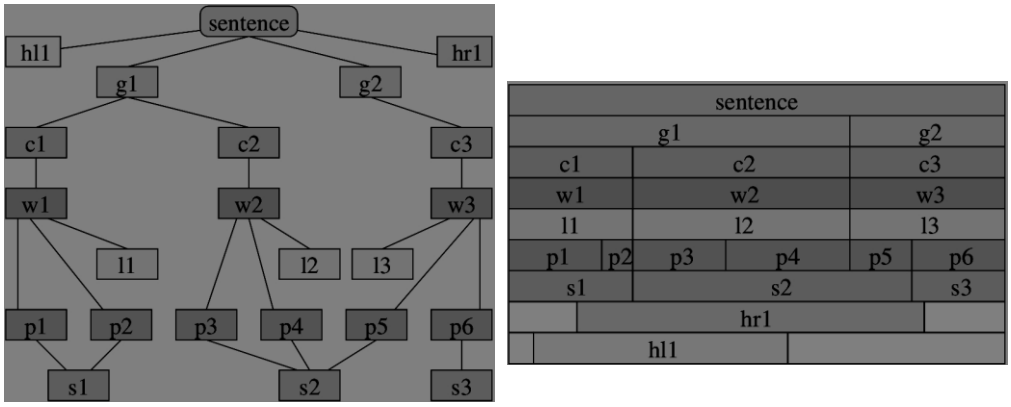

Figure 1: Example of annotations representation using graphs at left, and tiers at right.

As soon as the annotations are associated to a time representation, trees can easily and systematically be mapped into tiers. However, it is not always possible to map a hierarchy of tiers into a tree – it depends on the structure of the hierarchy, as shown below:

```
Parent tier:   Phonemes   | l  | a | n |  e  | l | a |
Child tier:    Tokens     | l' |   âne | est |   là  |

Parent tier:   Phonemes   | l  | a | n |  e  | l | a |
Child tier:    Syllables  |   l.a  |  n.e    |  l.a  |
```

Notice that there is no possibility of hierarchy between Tokens and Syllables. Consequently, the map from a representation to another one is something difficult but doable in many situations.

3 Annotation representation framework in SPPAS

Representing annotated data in SPPAS is of crucial importance for its automatic annotations, its analysis of annotations and for the software to be able to automatically annotate and analyze any kind of data files. Data representation of SPPAS is then commonly based on tiers, made of annotations. However, contrariwise to most of the existing annotation software tools, the time localization and label of each annotation are represented as complex objects to increase the availability of both: representing complex annotations and being compatible with other existing annotation representations. SPPAS also allows to represent uncertainty of annotations: it can concern the time precision (Bigi et al. 2014), the localization and/or the label (i.e. it can include alternative localizations and/or alternative labels). An *Annotation* in SPPAS is then made of:

- a *Location*, which is a list of *Localization* and its *score*. A *Localization* is either a *Point*, an *Interval* or a *Disjoint*. A *Point* includes a midpoint value and a radius to represent the vagueness of the localization (Bigi et al. 2014). *Disjoint* intervals are mostly used for dialog annotations;

- eventually a *Label* which is a list of *Tag* and its *score*. A *Tag* is either a string or a numerical value.

A *Tier* is then used to distribute the annotations of a recording. It can also contain a controlled vocabulary, a link to a primary media, etc. The definition of a tier proposed in (Brugman 2003) is then largely recovered: “a tier is a group of annotations that all describe the same type of phenomenon, that all share the same metadata attribute values and that are all subject to the same constraints on annotation structures, on annotation content and on time alignment characteristics.”. Tiers are stored into a *Transcription* object that may indicate a *Hierarchy* between tiers in the form of a list of *TimeAssociation* or *TimeAlignment* links. All the 3 objects *Transcription*, *Tier*, and *Annotation* can also contain metadata in the form *key:value* like for example: date:2016-12-01; annotator: John; etc.

A native format named XRA was developed to fit in such data representation. The physical level of representation of XRA obviously makes use of XML, XML-related standards and stand-off annotation. Due to an intuitive naming convention, XRA documents are human readable as far as possible within the limits of XML. The format specification as well as the texts encoded in XRA are “reusable, i.e., potentially usable in more than one research project and by more than one research team, and extensible, i.e., capable of further enhancement” (Ide and Brew 2000).

The proposed framework is implemented using the programming language Python. This Application Programming Interface is part of SPPAS and also distributed under the terms of the GNU Public License. The developers implement behaviors in abstract classes and each class is well-documented. Actually, it is quite easy to read some existing annotation file formats and to instantiate them into the proposed framework to use it (like exemplified in Section 4.3).

4 Converting files with SPPAS

4.1 Principle

SPPAS makes use of its internal data representation to convert files. A conversion then consists of two steps: First, the incoming file is loaded and mapped to the SPPAS data framework described in the previous section and second, such data will be saved to the expected format. These two steps are applied whatever the organization structure of annotations in the original or in the destination file format. This is illustrated by Figure 2 which includes the list of file formats and the corresponding software that are supported by version > 1.8 of SPPAS. Arrows are representing the following actions:

- *import from*, represented by a continued line with an arrow from the file format to the SPPAS framework;
- *partially import from*, represented by a dash line with an arrow from the file format to the SPPAS framework;

- *export to*, represented by an arrow from the SPPAS framework to the file format.

SPPAS version 1.8.0 supports Phonedit & Signaix (Teston et al. 1999), Praat (Boersma and Weenink 2001), HTK (Young and Young 1993), Sclite (NIST 2009), Xtrans (LDC 2005), Transcriber (Barras et al. 2001), Anvil (Kipp 2011), Elan (Wittenburg et al. 2006), Annotation Pro (Klessa 2016), subtitles and tab-separated files. The support of external formats is regularly extended to new formats by the author on-demand from the users and distributed to the community in the SPPAS regular updates (about ten a year).

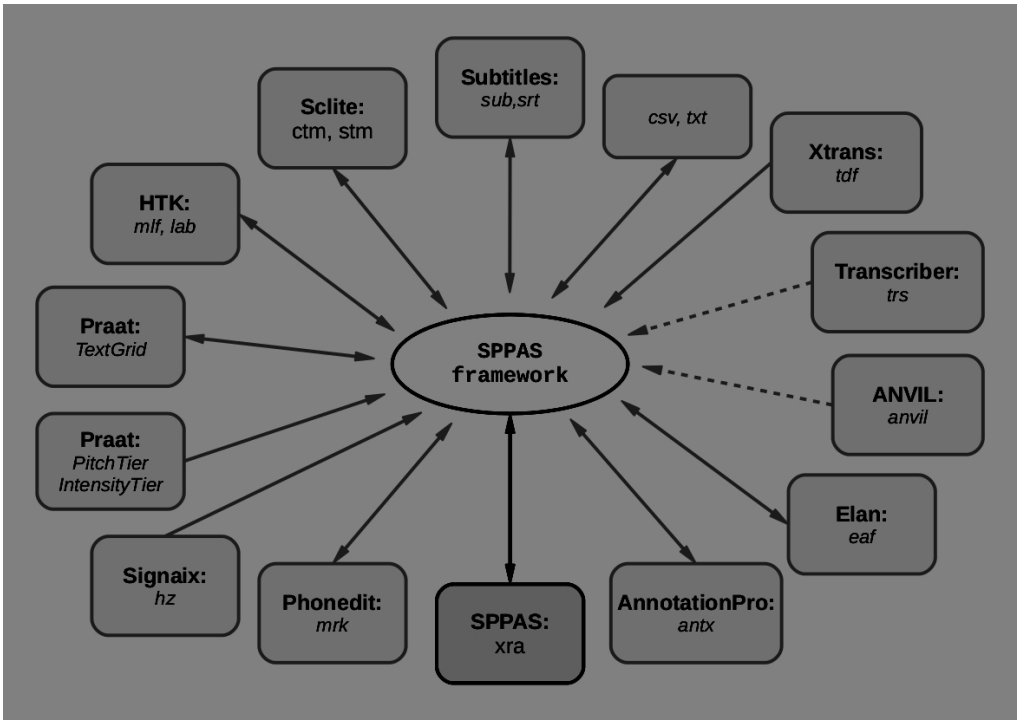

Figure 2: SPPAS conversion method and formats (version 1.8.0).

4.2 The given example

In the context of the International Workshop "Speech audio archives: preservation, restoration, annotation, aimed at supporting the linguistic analysis", a set of audio and TextGrid files were distributed. The file `Ayoreyo.TextGrid` will be used as example in the following.

TextGrid is the native format of *Praat*, a tool for manually annotating sound files. It provides different visualizations of audio data - waveform or spectrogram display - and, among other things, enables pitch contour as well as formant calculation and visualization. The Praat native files are in several native text formats; “TextGrid” is the one used to save annotated data.

To be used with SPPAS, it was first necessary to save the given file with UTF-8 encoding. There are 9 tiers with name TranscripAYO, TranslationENG, Words, WordsPhones,

Phones, Syllables, Morph-Syntax, language and Notes. We observe that tiers are used for both annotated data and metadata. Annotated data of tiers are represented as below:

```
intervals [1]:
    xmin = 0
    xmax = 0.1377517184116469
    text = "#"
intervals [2]:
    xmin = 0.1377517184116469
    xmax = 0.4928648872410315
    text = "chi"
intervals [3]:
    xmin = 0.4928648872410315
    xmax = 1.5253691515655043
    text = "#"
intervals [4]:
    xmin = 1.5253691515655043
    xmax = 3.747445456657043
    text = "siquere chise yocayode ore nani"
intervals [5]:
    xmin = 3.747445456657043
    xmax = 4.186875287771049
    text = "#"
intervals [6]:
    xmin = 4.186875287771049
    xmax = 8.13501133786848
    text = ""
```

Each annotation is represented by an interval, with 2 real numbers to represent respectively the beginning and the end of the interval, and a text. We can observe that the number of digits can increase up to 15. In our example, the audio file is sampled at 44,100 Hz, a frame is then during 0.000022676 seconds. This means that time value of an annotation in this TextGrid is about 100,000 times more precise that the framerate of the annotated recording.

4.3 Converting using SPPAS

There are several ways to use SPPAS: The Graphical User Interface (GUI), the Command-Line User Interface (CLI) and the Application Programming Interface (API).

There are two solutions to convert files from the GUI (Figure 3). The first one is directly from the File Explorer by selecting the file(s) to convert, clicking on the “Export” button and then choosing the file format from the given list. The second solution is the DataRoamer analysis component that is useful if the file as to be modified: renaming tiers, deleting tiers, changing their order, etc. In both cases, an export fails if:

- the given file is not in UTF-8 encoding;
- the given file contains several tiers and the output format supports only one;
- the given file contains corrupted annotations (as for example: two annotations of the same tier are starting and ending at the same time, the end of an interval is lower or equal to the beginning, etc).

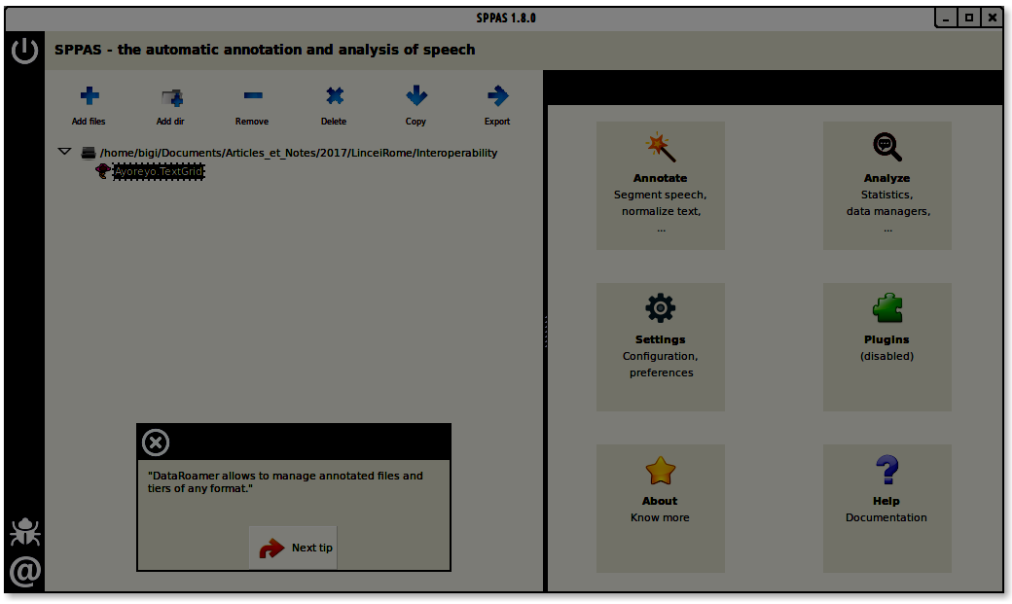

Figure 3: SPPAS main frame

Converting with the CLI is performed thanks to a Python script: `trsconvert.py`. It allows to select the tier(s) to export. Finally, the more powerful solution to convert files is the API but it requires some basic skills about Python programming language. Figure 4 is an example of the script that was specifically created for the purpose of this study. The first step is to load the file and to map it into the SPPAS framework. Then, the following optional modifications are performed: fixing hierarchy links between tiers, assigning a recording, replacing the last two tiers by metadata and fixing the language as metadata. The modified data representation is saved in `Ayoreyo.xra` file.

```
import sys
import os
SPPAS = os.getenv('SPPAS')
sys.path.append(os.path.join(SPPAS, 'sppas', 'src'))
from annotationdata.transcription import Transcription
from annotationdata.media import Media
import annotationdata.io

# Read input file
trs = annotationdata.io.read( "Ayoreyo.TextGrid" )

# Add "flexibility" in boundary time values
for tier in trs:
    tier.SetRadius(0.005)

# Optionnal: Set Hierarchy links
trs.GetHierarchy().addLink("TimeAlignment",trs.Find("Words"),trs.Find("TranscripAYO"))
trs.GetHierarchy().addLink("TimeAssociation",trs.Find("Words"),trs.Find("WordsPhones"))
trs.GetHierarchy().addLink("TimeAssociation",trs.Find("Words"),trs.Find("Morph-Syntax"))
trs.GetHierarchy().addLink("TimeAlignment",trs.Find("Phones"),trs.Find("Words"))
trs.GetHierarchy().addLink("TimeAlignment",trs.Find("Phones"),trs.Find("Syllables"))

# Optionnal: Set Media
m = Media('AyoreyoId', 'Ayoreyo.wav', 'audio/wav')
trs.AddMedia(m)
for tier in trs:
    tier.SetMedia(m)
```

```
# Optionnal: Set Metadata
trs.SetMetadata('iso639-3', u"ayo")
trs.SetMetadata('notes', u"# = pause; §§§ = unintelligible")
trs.Find("TranslationENG").SetMetadata("iso639-3", "eng")

# Optionnal: Remove/Modify the tiers that was used to declare metadata
trs.Pop()
trs.Pop()
trs.Find("TranscripAYO").SetName("Transcription")

# No set of a controlled vocabulary

# Write
annotationdata.io.write("Ayoreyo.xra", trs)
```

Figure 4: A Python script to convert the example file, with SPPAS version 1.8.0

An annotation of the resulting XRA file looks like the following one:

```
<Annotation>
  <Location>
    <Localization score="1.0">
      <Timeinterval>
        <Begin midpoint="1.5253691516" radius="0.005" />
        <End midpoint="3.7474454567" radius="0.005" />
      </Timeinterval>
    </Localization>
  </Location>
  <Label scoremode="max">
    <Text score="1.0">siquere chise yocayode ore nani</Text>
  </Label>
</Annotation>
```

As it was previously mentioned, the framework for annotations is very rich and contain several information like alternative labels or alternative localizations. The midpoint value was stored exactly as it is in the original file, which is useful to save back in a TextGrid file without any loss of information.

Either the original TextGrid file or the newly created XRA file can be converted in any of the supported formats; next section reports on some of them.

4.4 Example of results

Two exports into XML formats are supported: Elan and Annotation Pro. Elan export (.eaf) of the above mentioned annotation is:

```
<ANNOTATION>
    <ALIGNABLE_ANNOTATION ANNOTATION_ID="a3" TIME_SLOT_REF1="t3"
TIME_SLOT_REF2="t4">
      <ANNOTATION_VALUE>siquere chise yocayode ore nani</ANNOTATION_VALUE>
    </ALIGNABLE_ANNOTATION>
 </ANNOTATION>
```

Time values, in milliseconds, are stored in a list of time slots and annotations refer to such time slots. Only intervals are supported and overlaps are forbidden. Elan allows to set a controlled vocabulary, a hierarchy between tiers and a media. The other XML export supported by SPPAS

is the AnnotationPro native format (.antx). Each annotation is considered as a *Segment* of a previously defined *Layer*. Like for most of annotation tools, only intervals are supported and overlaps are forbidden. Each segment also contains graphical information used by the software to display it, like its colors. Finally, a segment can be enriched by 1 to 3 parameters.

Table 1 presents the other formats supported by SPPAS. CSV is not related to an annotation tool however it is supported by spreadsheet software. Phonedit (.mrk) is using the informal standard of INI configuration files: a simple text file with a basic structure composed of sections, properties and values. In that case, sections are used to represent tiers; then properties are the annotations identifiers for which a value is assigned: the label of the annotation. Only intervals are supported, however the degenerated interval is allowed and overlaps are also allowed. The time marked conversation scoring input (.ctm) is a native format of Sclite – a tool for scoring and evaluating the output of speech recognition systems. This file format is a concatenation of time mark records for each word in each channel of a waveform. The records are separated with a newline. Each word token must have a waveform id, channel identifier [A | B], start time, duration, and word text. Optionally a confidence score can be appended for each word. The segment time mark input file (.stm) is also a Sclite native format. The file consists of a concatenation of text segment records from a waveform file. Each record is separated by a newline and contains: the waveform's filename and channel identifier [A | B], the talker ids, begin and end times (in seconds), optional subset label and the text for the segment. SubRip files (.srt) contain formatted lines of plain text in groups separated by a blank line. The plain text is made of a numeric counter identifying each sequential subtitle, the time that the subtitle should appear on the screen, followed by --> and the time it should disappear and the subtitle text itself on one or more lines. Finally, the SubViewer files is a INI style file with a header section that contains metadata and rendering instructions. Immediately following is a [SUBTITLE] section, consisting of comma-delimited time ranges (accurate to one hundredth of a second) and a caption to be displayed during each range.

`Transcription,1.5253691516,3.7474454567,siquere chise yocayode ore nani`	.csv
`LBL_LEVEL_AA_000003= "siquere chise yocayode ore nani" 1525.369152 3747.445457`	.mrk
`Ayoreyo.wav A 1.5253691516 2.2220763051 siquere chise yocayode ore nani 1.0`	.ctm
`Ayoreyo.wav Transcription none 1.5253691516 3.7474454567 siquere chise yocayode ore nani`	.stm

`4`
`00:00:01,525 --> 00:00:03,747`
`siquere chise yocayode ore nani`
	.srt

`00:00:01.52,00:00:03.74` `siquere chise yocayode ore nani`	.sub

Table 1: Example of one annotation as it is represented in several file formats

5 Conclusion

Computational Linguistics make it necessary to produce and share reference annotations to which linguistic and computational models can be compared. The development of an expressive and generic framework for linguistic annotations is then very beneficial and many efforts are currently being made in this direction. Considerable attention has been devoted to the development of means to represent annotations so that phenomena at different levels can be merged and/or analyzed in combination. A particular attention has been on the development of standards for representing linguistic annotations.

This paper particularly focused on the problem of speech annotation representation framework and conversion. An XML file format was proposed, and a conversion tool was described and illustrated with an example. With such simple example, we observed that SPPAS framework is interoperable with a large number of file formats: like Elan, it supports hierarchy of annotations, controlled vocabularies and media assignment; like Annotation Pro, it supports an advanced media definition; like Phonedit, it allows overlaps of annotations; like Sclite, it supports alternative labels; like Praat, it allows to store all the 15 digits for time values, and it supports interval tiers and point tiers. Moreover, several other features already implemented in the framework will allow to support more formats in the future. Actually, the development of new file import/export is done by the author on-demand.

Some other components are also published within SPPAS software tool for the analysis of any kind of annotated files supported for importing by SPPAS. The DataStats analysis component is used to count occurrences of annotations and to estimate descriptive statistics as standard deviations, etc. It is also intended to be used to estimate the Kappa value in a near future. The DataFilter analysis component is used to extract annotations from tiers (Bigi and Saubesty 2015). With respect to searching, linguists are typically interested in locating patterns on specific tiers, with the possibility to relate different patterns of annotations on some tiers. DataFilter offers a powerful way to request/extract data, with the help of Allen's relations, without requiring any XML-related knowledge or skill. It has been already used in several studies as to find correlations between speech and gestures (Tellier et al. 2012), or to find which gestures are produced while pausing (Tellier et al. 2013), just to cite some of them. Finally, the use of the SPPAS framework in several studies or projects, and the user feedbacks allowed us to verify that the proposal of this paper is robust, useful and reliable.

SPPAS can be freely downloaded at: http://www.sppas.org.