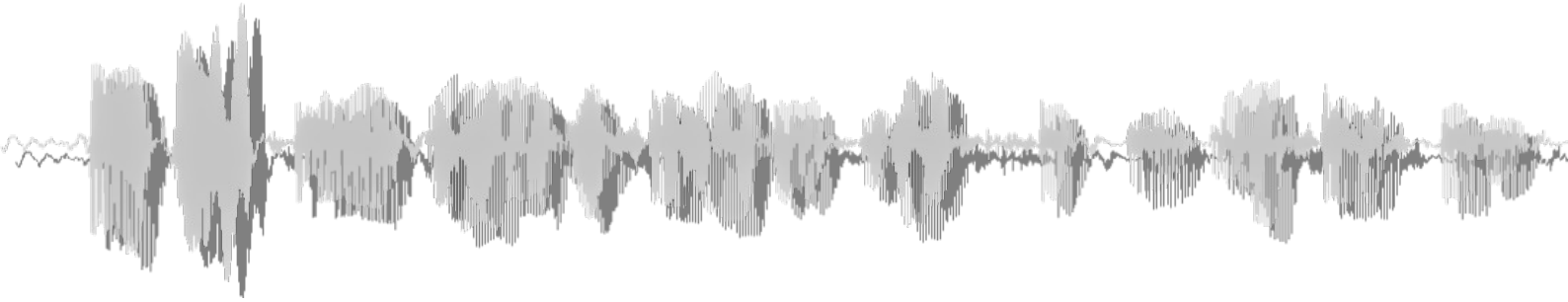

SPPAS annotation format

XRA

Schema version: 1.5

(Schema release date: November 2021)

Brigitte Bigi

contact@sppas.org

Table of content

1. Introduction

This document describes the structure of **XRA**, the SPPAS Annotation Format. An "xra" file contains XML elements and attributes of the XRA schema. SPPAS makes use of its internal data representation to represent transcriptions and "xra" is the format used for serializing these objects.

1.1 What is “Annotation” and why a new annotation format was required?

Corpus annotation “can be defined as the practice of adding interpretative, linguistic information to an electronic corpus of spoken and/or written language data. 'Annotation' can also refer to the end-product of this process” (Leech, 1997). Corpora are annotated with detailed information at various linguistic levels. In order to be useful for purposes such as qualitative or quantitative analyses, the annotations must be time-synchronized (time-aligned). Temporal information makes it possible to describe behavior or actions of different subjects that happen at the same time, and time-analysis of multi-level annotations can reveal levels of linguistic structures.

The process of annotating recorded speech is commonly made of 2 tasks: segmenting and labelling. Labelling determines the transcriptions of a recording. Segmenting consists in pairing these transcriptions with time-stamps to indicate the localization and thus the length of the transcibed labels in the recording. Figure 1 is illustrating these kind of annotations grouped into tiers.

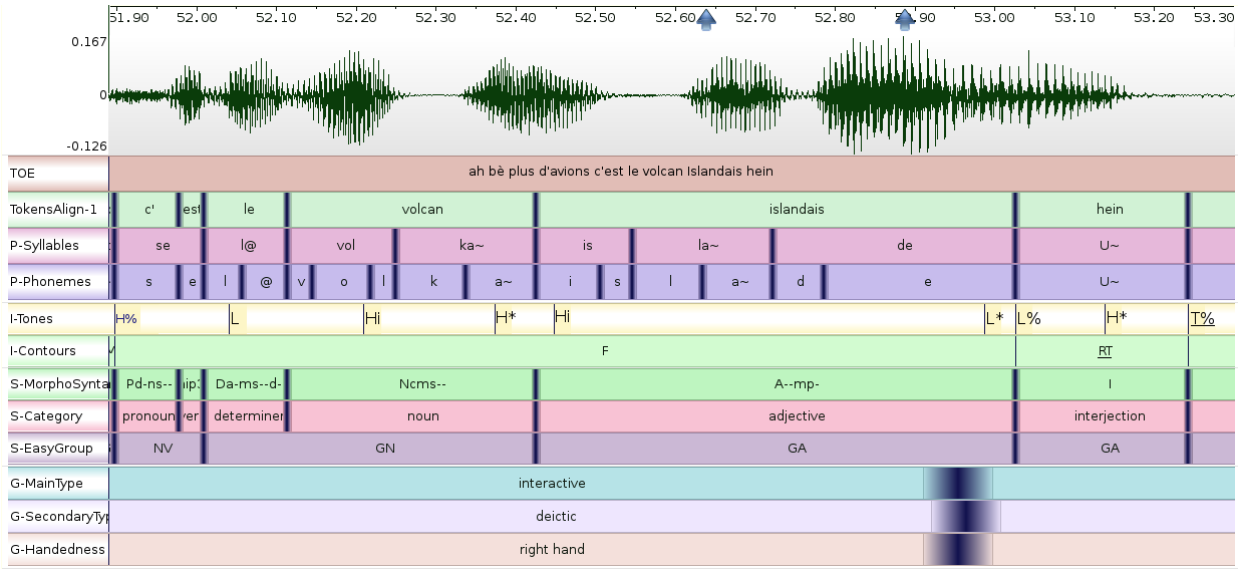

Figure 1: Example of a timeline with annotated tiers

It is a common practice in Corpus Linguistics to admit that **annotating is an inherently ongoing process**. Linguistics data are annotated several times by one or several annotators, each one annotates according to his/her knowledge, beliefs and uncertainty. On the one hand, some annotations can be assigned without any doubt; and this clearly indicates that some properties about the field exist. On the other hand, it is the fundamental nature of research to deal with hard-to-annotate phenomena that causes indeterminacy during the annotation

process. Most of automatic annotation systems includes a decision-making procedure to deliver a final result, based on scores assigned to a set of possible annotations – either segmentations or labels. The score often corresponds to the reliability of a solution according to a model. Close scores could be interpreted as an uncertainty of the decision-making procedure according to a given model. If some annotation does not easily fit into an existing theory, it is likely to be something linguistically interesting and worthy of attention. For automatic annotations, modelling variances and indeterminacy of an automatic system will reduce the ambiguity in interpreting and applying the resulting annotations. For manual annotations, keeping information about the annotator's uncertainty is motivated by the fact that until the annotator reaches a decision, he/she is likely to leave the data un-annotated, so an uncertain annotation is better. It then allows other annotators to share experience about this annotation. Perhaps a similar hard-to-annotate concept has been encountered by another annotator. In some cases, sharing experiences about uncertain annotations can lead to certainty. In some other cases, annotating uncertainty will highlight the fact that a concept is inherently indeterminate.

The suggestion that a concept can be adequately represented only by an interval and a single label value information is false, as is well-known to anyone who has had to model both time and label. The Annotation framework therefore needs to allow the representation of uncertainties, for these annotations to become part of the framework itself. The proposed scheme allows to represent **imprecision** (see *Point* element definition, the last 3 tiers of Figure 1) and **indeterminacy** (see *Tag* element definition).

1.2 A few global information about xra

Most of the elements of the scheme have an ‘id’ attribute. SPPAS is using a fully randomly generated Global Unique Identifier but there’s no limitation and any string is accepted as soon as it is a unique identifier in the document.

A few constraints on tiers are:

- the localization of the annotations are either of type point or interval but not a mix of them;
- a mix of different types of label tags is not allowed;
- the xra file is UTF-8 encoding.

However, the followings are supported:

- the annotations can overlap;
- the annotations can have a serie of labels, each one with alternative tags;
- the tags of a label can be of type: string, integer, float, boolean or point.
- there's no limit on the metadata that can be added to the transcription file, the tiers, and the annotations;
- etc...

2. Simplified UML class diagrams

Figure 2 shows the simplified UML class diagram of the XRA schema for an annotation. An annotation is a complex object in order to:

- represent the largest linguistic phenomena as possible;

- to be able to import/export annotated files of most of the other software tools;
- to deal with uncertainty. Uncertainty is a state of having limited knowledge where it is impossible to exactly describe the existing state, a future outcome, or more than one possible outcome.

An annotation is then made of an optionnal sorted list of labels and a required location. A label is a set of possible tags, with their scores. An annotation is requiring at least a location. The localization of an annotation is either a point, an interval or a list of intervals (actually it'is not used into SPPAS but the scheme allows it).

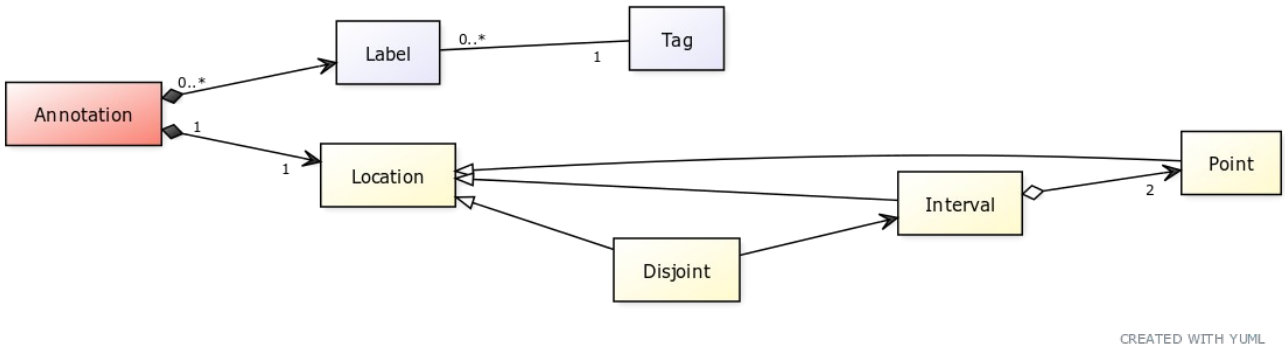

Figure 2: Simplified representation of an annotation

Figure 3 shows a simplified UML class diagram of the XRA schema. A document contains 4 elements: a list of media, a list of tiers, a list of hierarchy links among tiers and a list of controlled vocabularies. Annotations described in Figure 1 are contained in tiers. Except the hierarchy, all of these elements contains metadata.

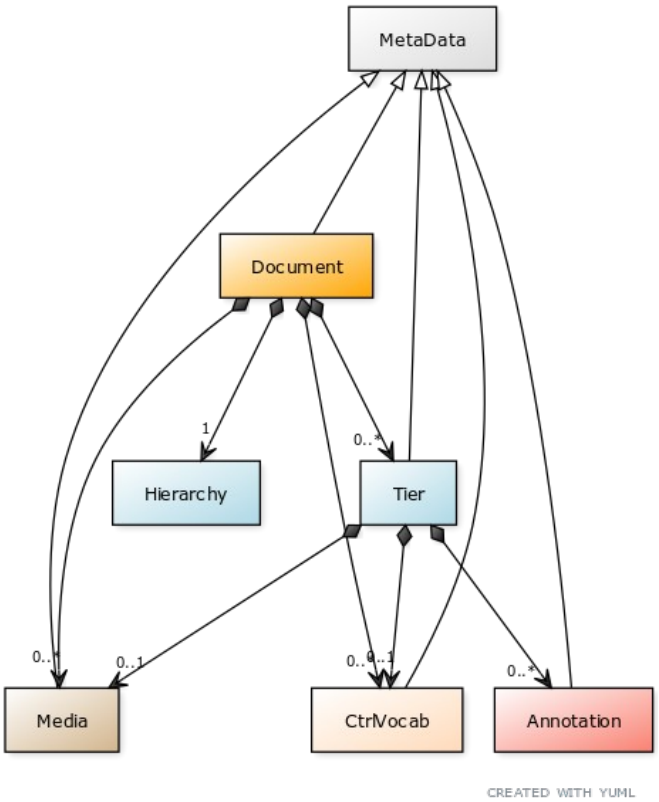

Figure 3: Simplified representation of a document

3. Scheme of an annotation

3.1 The annotation element

The *Annotation* element defines an annotation that is associated with a segment of a media by means of references to it's location. It is the result of the segmentation task.

The *Annotation* element can also contain a list of *Label* elements which are the result of the labelization task.

Annotation scheme

```
<xsd:complexType name="annotationType">
  <xsd:sequence>
    <xsd:element name="Metadata" type="metadataType" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="Location" type="annotationLocation" minOccurs="1" maxOccurs="1" />
    <xsd:element name="Label" type="annotationLabel" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="id"    type="xsd:ID"     use="required"/>
  <xsd:attribute name="score" type="xsd:double" use="optional"/>
</xsd:complexType>
```

Annotation example

```
<Annotation id="0ccfe8c8-7591-4d1d-b556-e64766e94fed">
  <Metadata>
    <Entry key="annotator_comment">hum… something important here!</Entry>
  </Metadata>

…
</Annotation>
```

3.2 The annotation location

The location of an annotation is the result of the segmentation task. The Location element is only a container for the appropriate localization in the recording, then it's a choice among *Point* element, *Interval* element or *Disjoint* element.

Location scheme

```
<xsd:complexType name="annotationLocation">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="Point" type="pointType" minOccurs="1" maxOccurs="unbounded"/>
      <xsd:element name="Interval" type="intervalType" minOccurs="1" maxOccurs="unbounded"/>
      <xsd:element name="Disjoint" type="disjointType" minOccurs="1" maxOccurs="unbounded"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```

3.1.1 Point

The *Point* element is representing a point in the timeline with both a midpoint value and a radius value. The radius is representing half of the vagueness of the point. See [1] for details and next Figure. It allows the human annotator to annotate the localization of an interval as "the annotation starts about here, and ends about there". It may in some cases prevent the analyst from drawing wrong

conclusions unsupported by the original data. It also allows the automatic annotations to indicate their precision. For example, the “Alignment” is based on the analysis of an audio file with windows of 10ms and the result is a segmentation starting and ending within theses windows.

Even if the scheme allows to mix types, into SPPAS the midpoint and the radius must be of the same type: either integer or float. Moreover, a tier must have all its annotated points of the same type.

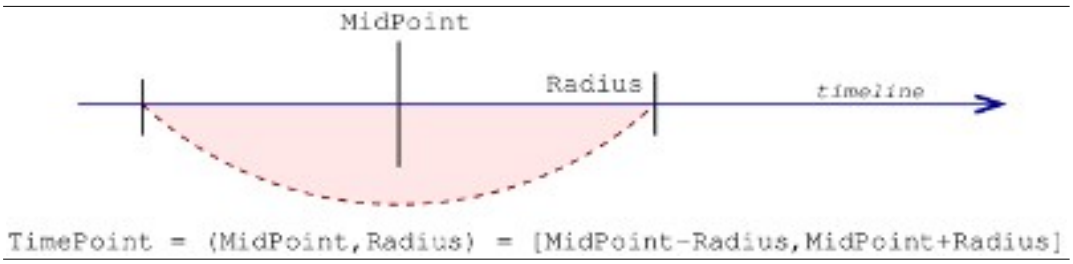

Figure 3: Representation of a point in SPPAS and in the XRA scheme

Point scheme

```
<xsd:complexType name="pointType">
    <xsd:attribute name="midpoint" use="required"/>
    <xsd:attribute name="radius" use="optional"/>
    <xsd:attribute name="score" type="xsd:double" use="optional"/>
</xsd:complexType>
```

Point examples

```
<Point score="1.0" midpoint="0.1234" radius="0.001" />
<Point score="0.8" midpoint="12" radius="1" />
```

3.1.2 Interval

An interval is a container for a Begin element and a End element. Both are points. In SPPAS, the degenerated interval is not allowed: both points can’t represent the same value in the timeline.

Interval scheme

```
<xsd:complexType name="intervalType">
    <xsd:all>
        <xsd:element name="Begin" type="pointType" minOccurs="1" maxOccurs="1" />
        <xsd:element name="End" type="pointType" minOccurs="1" maxOccurs="1" />
    </xsd:all>
    <xsd:attribute name="score" type="xsd:double" use="optional"/>
</xsd:complexType>
```

Interval example

```
<Interval score="1.0">
    <Begin midpoint="0.1234" />
    <End midpoint="0.3600" radius="0.005" />
</Interval>
```

3.1.3 Disjoint

Unused in current version, disjoint intervals were introduced in the scheme for

discourse annotations which can refer to several alternative locations. For example, it could be useful when annotating the reference of a pronoun: a pronoun is refering to only one noun location but several ones can be candidates and then it can have several alternative locations, one for each canditate.

3.3 The annotation labelling

Labels of an annotation is the result of the labelization task. An annotation can contain a list of Label elements. This list is for example the result of the text normalization process which includes a segmentation in tokens of the orthographic transcription but at this stage there's no anchor of such individual tokens in the timeline. A *Label* element is a container for a list of alternative tags. Optionnally, a label can be identified with a key.

Label scheme

```
<xsd:complexType name="annotationLabel">
   <xsd:sequence>
      <xsd:element name="Tag" type="annotationTag" minOccurs="0" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name="tiername" type="stringType" use="optional"/>
</xsd:complexType>
```

The *Tag* element has an optionnal "score" attribute. By default, the type of a tag is a string. However, contrariwise to all the other annotation formats, a tag can be an integer value, a float value, a boolean value, a point – i.e. (x, y, r) coordinates, or a rectangle - i.e. (x, y, w, h, r), where "r" is a radius. Notice that in SPPAS, a tier must have all its annotated tags of the same type.

Tag scheme

```
<xsd:complexType name="annotationTag">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
       <xsd:attribute name="score" type="xsd:double" use="optional"/>
       <xsd:attribute name="type" type="annotationTagType" use="optional"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<xsd:simpleType name="annotationTagType">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="float|int|bool|str|point|rect"/>
  </xsd:restriction>
</xsd:simpleType>
```

Tag examples

```
<Label>
    <Tag score="0.9">noise</Tag>
    <Tag score="0.1" />
</Label>
<Label>
    <Tag score="0.8" type="str">The best tag of the annotation</Tag>
    <Tag score="0.2" type="str">The west flag of the anotation</Tag>
</Label>

<Label>
   <Tag>D-@</Tag>
   <Tag>D-i:</Tag>
   <Tag>D-V</Tag>
</Label>
```

Tag scheme

```
<Label>
   <Tag>f-l-aI-t</Tag>
</Label>



<Label>
   <Tag score="0.604">D-@</Tag>
</Label>
```

4.1 Examples of the Annotation element

Example 1: Tagging an image of a video with 2 different points

```
<Annotation id="av1">
    <Location>
         <Interval>
              <Begin midpoint="0.120" />
              <End midpoint="0.160" />
          </Interval>
    </Location>
    <Label>
        <Tag type="point">(234,402,12)</Tag>
    </Label>
    <Label>
        <Tag type="point">(256,802)</Tag>
    </Label>
</Annotation>
```

Example 2: Tier “PronTokAlign”, result of the “Alignment” automatic annotation

```
<Annotation id="75fc2cd3-9cbf-4f05-9b4f-bddd2eb3d4da">
    <Location>
      <Interval>
         <Begin midpoint="2.045" radius="0.005" />
         <End midpoint="2.305" radius="0.005" />
      </Interval>
    </Location>
    <Label>
      <Tag score="0.464">w-@-z</Tag>
    </Label>
</Annotation>
```

4. Scheme of the document

4.1 Element Document

This is the root element of an "xra" document. It contains all other elements (Figure 2) and have four optional attributes:

- date - the creation date;
- author - the person or program that created the file;
- name - an identifier or any other relevant information;
- format - by convention it's the version format of the scheme (1.5).

Document scheme

```
<xsd:element name="Document">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Metadata"type="metadataType" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="Media"    type="mediaType" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="Tier"     type="tierType" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="Hierarchy"  type="hierarchyType" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="Vocabulary" type="vocabType" minOccurs="0" maxOccurs="unbounded"/
>
    </xsd:sequence>
    <xsd:attribute name="author" type="stringType" use="optional"/>
    <xsd:attribute name="date"   type="xsd:date"   use="optional"/>
    <xsd:attribute name="format" type="stringType" use="optional" default="1.5"/>
    <xsd:attribute name="name"   type="stringType" use="optional"/>
  </xsd:complexType>
</xsd:element>
```

Document example

```
<?xml version='1.0' encoding='utf-8'?>
<Document author="SPPAS 4.0 (C) Brigitte Bigi"
          date="2021-10-21T07:31:45+02:00"
          format="1.5"
          name="Alignment">
…
</Document>
```

4.2 Element Metadata

There should be at most one *Metadata* element in the document; it is a container for the *Entry* element. Metadata is a general purpose element to store pairs of (key,value) entries. It then allows to store any kind of information, without limitation. It is then an opportunity to save valuable information in the xra file, for example about the annotator, the speaker, the recording condition or anything else. When saving a transcription object, SPPAS is using this element to save it's name, version, url, etc. The *Metadata* element is also used to indicate:

- the language with it's iso859-3 code, it's name and url;
- the license;
- the version of the file is useful if the file is annotated several times.

Metadata scheme

```
<xsd:complexType name="metadataType">
  <xsd:sequence>
    <xsd:element name="Entry" type="metaEntryType" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="metaEntryType">
    <xsd:simpleContent>
        <xsd:extension base="stringType">
            <xsd:attribute name="key" type="stringType" use="required"/>
        </xsd:extension>
    </xsd:simpleContent>
</xsd:complexType>
```

Metadata example

```
<Metadata>
```

Metadata scheme

```
   <Entry key="id">5881433d-8644-404f-af90-037d7bed7655</Entry>
   <Entry key="software_name">SPPAS</Entry>
   <Entry key="software_version">4.0</Entry>
   <Entry key="software_url">http://www.sppas.org/</Entry>
   <Entry key="software_author">Brigitte Bigi</Entry>
   <Entry key="software_contact">contact@sppas.org</Entry>
   <Entry key="software_copyright">Copyright (C) 2011-2021 Brigitte Bigi</Entry>
   <Entry key="language_iso">iso639-3</Entry>
   <Entry key="language_code_0">fra</Entry>
   <Entry key="language_name_0">French</Entry>
   <Entry key="language_url_0">https://iso639-3.sil.org/code/fra/</Entry>
   <Entry key="file_license_text_0">GNU GPL V3</Entry>
   <Entry key="file_license_url_0">https://www.gnu.org/licenses/gpl-3.0.en.html</Entry>
   <Entry key="file_name">F_F_B003_P8-palign.xra</Entry>
   <Entry key="file_version">1</Entry>
</Metadata>
```

4.3 Element Tier

The *Tier* element is a container for a list of *Annotation* elements, decribed in the previous section. A tier has a required ‘id’ attribute. The name of the tier is optionnal.

Tier scheme

```
<xsd:complexType name="tierType">
  <xsd:sequence>
    <xsd:element name="Metadata" type="metadataType" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="Annotation" type="annotationType" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="id"       type="xsd:ID"     use="required"/>
  <xsd:attribute name="tiername" type="stringType" use="optional"/>
</xsd:complexType>
```

Tier example

```
<Tier id="fc753315-2249-4be2-bef3-181aa4b65569" tiername="PhonAlign">
  <Metadata>
    <Entry key="annotator_name">My name</Entry>
    <Entry key="annotator_level">expert</Entry>
    <Entry key="annotator_comment">tier manually verified</Entry>
  </Metadata>
</Tier>
```

4.4 Element Media

The XRA scheme was elaborated to save information about annotations. Most of the time these annotations are discribing a linguistic phenomenum which occurs or which is related to some “recording”. A *Media* element allows to define a media file annotations of a tier refer to. The document can contain as many *Media* elements as needed. They are commonly containing the list of tiers which the annotations are refering to.

A Media element has 2 required attributes – an identifier and an url, and an optional one – the mime type. The url attribute is the absolute name of the media file – i.e. including it’s path, filename and extension.

Media scheme

```
<xsd:complexType name="mediaType">
  <xsd:sequence>
    <xsd:element name="Metadata" type="metadataType" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="Tier" type="idTierType" minOccurs="0" maxOccurs="unbounded" />
    <xsd:element name="Content"  type="stringType" minOccurs="0" maxOccurs="1" />
  </xsd:sequence>
  <xsd:attribute name="id"        type="xsd:ID"     use="required"/>
  <xsd:attribute name="url"       type="xsd:anyURI" use="required"/>
  <xsd:attribute name="mimetype"  type="stringType" use="optional"/>
</xsd:complexType>
```

Media example

```
<Media id="cc1f0c99-8224-457a-bdef-609b4aedecb1" url="\PATH\F_F_B003_P8.wav"
mimetype="audio/wav">
   <Metadata>
      <Entry key="framerate">44100</Entry>
   </Metadata>
   <Tier id="fc753315-2249-4be2-bef3-181aa4b65569" />
   <Tier id="dcecf8e5-0c89-4b87-a0bf-81654e2b84b1" />
   <Tier id="ab30f77e-3e07-42a4-af14-4a1b35151a94" />
</Media>
```

4.5 Element Hierarchy

A hierarchy can be used to fix constraints on the localization of annotations in tiers. These constraints can be of 2 types:

- TimeAssociation: the points of a child tier are all equals to the points of a reference tier, as for example:

```
| parent:  Words     | l' |  âne  | est  |  là   |
| child:   Lemmas    | le |  âne  | être |  là   |
```

- TimeAlignment: the points of a child tier are all included in the set of points of a reference tier, as for example:

```
| parent: Phonemes   | l  | a | n |  e  | l | a |
| child:  Words      | l' |  âne  | est |  là   |
```

and:

```
| parent: Phonemes   | l  | a | n |  e  | l | a |
| child:  Syllables  |   l.a  |  n.e   |   l.a |
```

In these examples, notice that there's no hierarchy link between "Words" and "Syllables" because some syllables can overlap 2 words and notice that "Phonemes" is the grand-parent of "Lemmas".

And the following rules are applied:

1. A child can have ONLY ONE parent;
2. A parent can have as many children as wanted;
3. A hierarchy is a tree, not a graph.

The TODO list includes a “partial” time association, i.e. to consider a time association that is not fully completed:

```
| parent:  Tokens | l' |  âne  | euh | euh | est  |  là   | @ |
| child:   Lemmas | le |  âne  |           | être |  là   |
```

Hierarchy scheme

```
<xsd:complexType name="hierarchyType">
  <xsd:sequence>
    <xsd:element name="Link" type="linkType"  minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="linkType">
  <xsd:attribute name="type"   type="stringTierRelation" />
  <xsd:attribute name="from"   type="xsd:IDREF" use="required" />
  <xsd:attribute name="to"     type="xsd:IDREF" use="required" />
</xsd:complexType>
<xsd:simpleType name="stringTierRelation">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="TimeAlignment|TimeAssociation"/>
  </xsd:restriction>
</xsd:simpleType>
```

Hierarchy example

```
<Hierarchy>
      <Link type="TimeAlignment" from="t2" to="t3" />
      <Link type="TimeAssociation" from="t2" to="t4" />
      <Link type="TimeAssociation" from="t4" to="t5" />
</Hierarchy>
```

4.6 Element Vocabulary

A controlled Vocabulary is a set of tags. It is used to limit the use of tags in a label: only the accepted tags can be set to a label. A *Vocabulary* element is a container for a list of *Entry* elements. It has a required 'id' attribute and an optionnal 'description' one. The vocabulary contains also the list of tiers it refers to. In the following exemple, the vocabulary defining the only 3 accepted tags for a smile annotation: 0, 1 or 2.

Vocabulary scheme

```
<xsd:complexType name="vocabType">
  <xsd:sequence>
    <xsd:element name="Entry" type="vocabEntryType" minOccurs="1" maxOccurs="unbounded" />
    <xsd:element name="Tier" type="idTierType" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:IDREF" use="required" />
  <xsd:attribute name="description" type="stringType" use="optional" />
</xsd:complexType>

<xsd:complexType name="vocabEntryType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="type" type="annotationTagType" use="optional"/>
      <xsd:attribute name="description" type="stringType" use="optional" />
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

Vocabulary example

```
<Vocabulary id="v0" description="Smile intensity scale">
    <Entry type="int" description="No smiling at all">0</Entry>
    <Entry type="int" description="Is smiling">1</Entry>
    <Entry type="int" description="Is laughting">2</Entry>
    <Tier id="t2"/>
</Vocabulary>
```

5. Full scheme and sample

The SPPAS package includes the full description of the XRA scheme (.xsd) and a sample file (.xra). Both are located in the “sppas/etc/xml” folder of the package.

The API of the framework representing these objects is in the package “anndata” of SPPAS (“`sppas/src/anndata`” folder). The file “`sppas/src/anndata/aio/xra.py`”) is the implementation of the import/export of “.xra” files from/to the framework.

6. References

[1] Brigitte Bigi, Tatsuya Watanabe, Laurent Prévot (2014). **Representing Multimodal Linguistic Annotated Data**. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pp. 3386-3392, Reykjavik, Iceland.

[2] Brigitte Bigi (2018). **Annotation representation and file conversion tool**. *Contributi del Centro Linceo Interdisciplinare ‘Beniamino Segre’ (ISSN 0394-0705), 137,* pp. 99-116.