

# Interoperability of Multimodal Annotation Tools

## Report for EXMARaLDA

thomas Schmidt, Thomas.schmidt@uni-hamburg.de

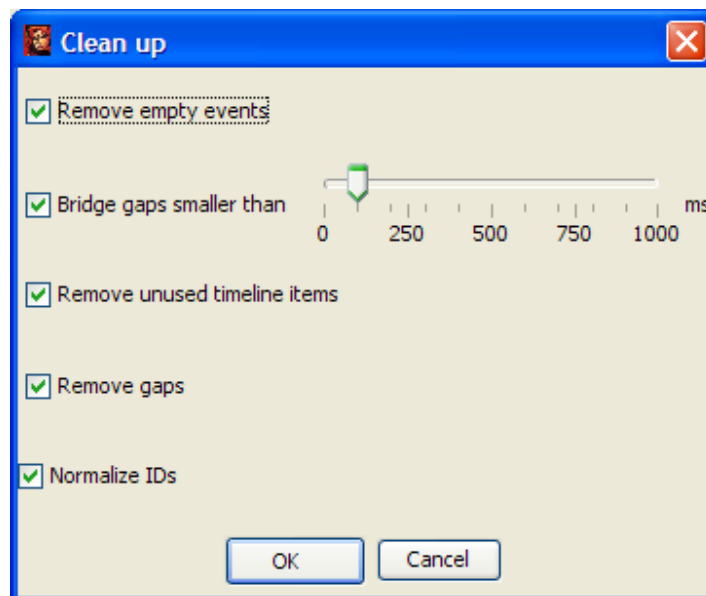
### EXMARaLDA to AG export

The export is done through the XSL stylesheet `exmaralda2ag.xsl`. EXMARaLDA's tier meta-information (ID, category, type, speaker and display name) are stored in the `MetadataElements`. EXMARaLDA timeline-items are mapped onto AG Anchors; if there are timeline items without absolute timestamps, the exporter calculates timestamps by interpolating between the nearest timeline items that do have timestamps. If the timeline does not have any timestamps at all, dummy offsets (i.e. '1.0' for the first item, '321.0' for the 321<sup>st</sup> etc.) are used. Offsets are converted from floating point seconds to integer milliseconds. EXMARaLDA events are mapped onto AG annotations. Every annotation has exactly one Feature element whose name is always 'description' and which contains the annotation string.

### AG to EXMARaLDA import

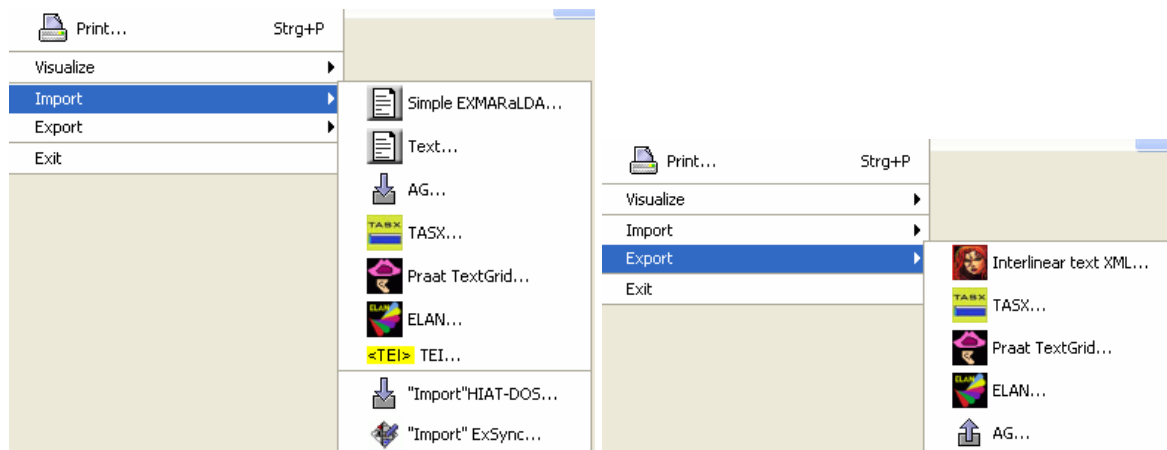
The import is done through the XSL stylesheet `ag2exmaralda.xsl`. I had to use XSLT version 2 here, because the algorithm requires grouping annotations according to their type attribute and to the names of their Feature elements, and this feature is not available in XSLT 1. The stylesheet proceeds as follows: For every available combination of type attribute in the Annotation element and name attribute in the Feature element, it constructs an EXMARaLDA tier. This tier is then filled with all the annotations of the given combination. EXMARaLDA's timeline is constructed by collecting all Anchor elements, ordering them by their offsets and mapping them onto EXMARaLDA's timeline item element. This can cause problems when there are anchors with identical offsets (see below).

After the import is completed, EXMARaLDA offers a clean-up option with which users can remove empty events, bridge small gaps in the timeline, remove unused timeline items, remove gaps in the timeline and normalize IDs (all of this is optional).



## Integration into the EXMARaLDA Partitur-Editor

The import and export functions have been integrated into the EXMARaLDA Partitur-Editor and will be made available in the upcoming version 1.3.4.



The editor has had an AG export function for quite some time, because I thought that users might want to reuse data with the AG toolkit (AG TableTrans, specifically). It seems that nobody actually uses that function, so I replaced it with the new version even though the result can now not be read with TableTrans. I will keep the ELAN import and export functions also in future versions because they take care of some things (e.g. speaker assignment) which the AG import/export between EXMARaLDA and ELAN does not (yet) handle.

## Some remarks on my choice of technology

Since almost all multi-modal annotation formats are XML formats, it seemed like a natural choice to do the conversion with XSL stylesheets. This has several advantages:

- 1) the conversion routines are in the same (standardized, widely used) meta-language as the data (i.e. XML)
- 2) Using XSL makes one independent of changing, possibly buggy specialised libraries (e.g. the AGLIB).
- 3) the conversion routines remain compact and therefore easy to maintain (because XSL has a declarative syntax – all the parsing, processing, etc. is left to the stylesheet processor);
- 4) XSL stylesheets can be integrated with relatively little effort into larger software packages (i.e. the tools themselves). There are ready-to-use XSL libraries for Java and, I think, for every other major programming language. Doing a stylesheet conversion is usually just a matter of two or three lines of code with calls to that package.

## Some observations on AG

Among other things, our effort has revealed some difficulties with the annotation graph framework. I think it's worth making a note of these:

- 1) AGLIB writes AG data which does not conform to the AG DTD. The error is that it uses attributes “startAnchor” and “endAnchor” where the DTD simply requires “start” and “end”.
- 2) AGLIB does not allow adding MetaData in some places of an AG file although the DTD explicitly allows it there.
- 3) The method of constructing qualified identifiers for anchors, annotations etc. by using the colon conflicts with the way XML defines for the use of namespaces. This has caused trouble in validation. If a different separator character (underscore) is used, the files validate, but cannot be read anymore by AGTK tools such as TableTrans.
- 4) Similarly, the use of namespaces has caused some trouble in the XSL transformations. As far as I can see, namespace awareness is only needed for the xlink:href attribute. If there ever is a version 1.2. of the DTD, it might be worth considering if the namespace for this one attribute is really worth all the trouble it is causing.

## **Future work**

Concerning the EXMARaLDA import/export, I will look into the problem of anchors with identical timestamps. I will also use our AG format as an export option in the corpora we’re compiling at our institute (presently, TEI is the only option).

Concerning the interoperability between tools in general, I think that a good next step might be to find an agreement on how to encode additional information about tiers in the AG file. For example, the information about tier dependencies might be exchanged between ANVIL and ELAN, if we agree to always name that feature “tier\_dependency” in the tier meta-information part of the AG file. Likewise, information about speaker assignment could be exchanged between EXMARaLDA and ELAN by agreeing on a feature named “speaker”. What is good about our present solution is that it allows the export routine to simply make that information available and to let the individual tool import routine decide whether or not it can make use of it. We might have another look at the comparison matrix and see which additional tier features are candidates for being exchanged between tools.

## **Tests**

I ran a last test on all the files that other tools exported. All these test files come from [http://multimodal-annotation.org/public/projects/multimodal/converted\\_files](http://multimodal-annotation.org/public/projects/multimodal/converted_files)

### MacVissta

*20070526\_316315582\_to\_ag\_230047.xml / 20070526\_784400475\_to\_ag\_230012.xml*

The files can not be imported because the tier assignment is still encoded in the `<Feature name="tier">` element rather than in the type attribute of the Annotation element. Travis says he’ll change this.

### Transformer

*pear\_ELAN\_2AG.xml / pear\_PRAAT\_2AG.xml*

The files can be imported without problems.

*pear\_EXMARaLDA\_2AG.xml*

The file can be imported, but for one tier (with ID `SPK0[X]-d-nv`), every Annotation seems to be present twice. EXMARaLDA complains about this and then distributes the Annotations onto two separate tiers.

## ANVIL

*fromAnvil\_conversation.xml / fromAnvil\_pear.xml*

I had to change my conversion routine because the Anvil AG export does not output the anchors in temporal order (the DTD does not require this). When I take this into account, the files can be imported, but there are some overlapping events in the tiers with ID `speaker_1.speech.shorttext` (about which EXMARaLDA complains and then distributes them onto two tiers). For example, in the pear story file, the following annotations overlap (according to the EXMARaLDA definition of overlap):

```
<Annotation id="anvil:10" type="speaker_1.speech.shorttext" start="anvil:8" end="anvil:9">
  <Feature name="praatwords">starts
</Feature>
</Annotation>
<Annotation id="anvil:13" type="speaker_1.speech.shorttext" start="anvil:11" end="anvil:12">
  <Feature name="praatwords">out</Feature>
</Annotation>
```

The corresponding anchors are:

```
<Anchor id="anvil:8" offset="361.059993" unit="ms" signals="anvil:1:Timeline1:Signal1"/>
<Anchor id="anvil:11" offset="725.4899740000001" unit="ms" signals="anvil:1:Timeline1:Signal1"/>
<Anchor id="anvil:9" offset="725.4899740000001" unit="ms" signals="anvil:1:Timeline1:Signal1"/>
<Anchor id="anvil:12" offset="1083.67002" unit="ms" signals="anvil:1:Timeline1:Signal1"/>
```

It seems the Anvil export is defining two distinct anchors with identical offsets, and my import puts these into the timeline in the wrong order because there is no way of knowing what the right order is. This may be a general problem with importing data from tools *without* an explicit timeline into tools *with* an explicit timeline (see the comparison matrix). I don't think we discussed this anywhere and I'm not sure how to handle this. Maybe I should map anchors with identical offsets in AG to one and the same timeline item in EXMARaLDA. I'll give this a try some time soon.

## ELAN

*pear.fromElan.ag.xml*

The file can be imported without problems.

## EXMARaLDA

*PearStory\_AG\_from\_EXMARaLDA.xml*

The file can be imported without problems (surprise, surprise!)