Published in: Schmidt, Thomas; Duncan, Susan; Ehmer, Oliver; Hoyt, Jeffrey; Kipp, Michael; Loehr, Dan et al.: An exchange format for multimodal annotations. In Michael Kipp (Ed.): Multimodal corpora. From models of natural interaction to systems and applications. Berlin [u.a.]: Springer, pp. 207–221.

POSTPRINT

An exchange format for multimodal annotations

Thomas Schmidt, University of Hamburg
Susan Duncan, University of Chicago
Oliver Ehmer, University of Freiburg
Jeffrey Hoyt, MITRE Corporation
Michael Kipp, DFKI Saarbrücken
Dan Loehr, MITRE Corporation
Magnus Magnusson, Human Behavior Laboratory Reykjavik
Travis Rose, Virginia Tech
Han Sloetjes, MPI for Psycholinguistics Nijmegen

Abstract This paper presents the results of a joint effort of a group of multimodality researchers and tool developers to improve the interoperability between several tools used for the annotation and analysis of multimodality. Each of the tools has specific strengths so that a variety of different tools, working on the same data, can be desirable for project work. However this usually requires tedious conversion between formats. We propose a common exchange format for multimodal annotation, based on the annotation graph (AG) formalism, which is supported by import and export routines in the respective tools. In the current version of this format the common denominator information can be reliably exchanged between the tools, and additional information can be stored in a standardized way.

Acknowledgments The initiative described in this paper was sponsored in part by funding from the MITRE Corporation Technology Program. We also gratefully acknowledge the Annotation Graph and Annotation Graph Toolkit projects.

1. Introduction

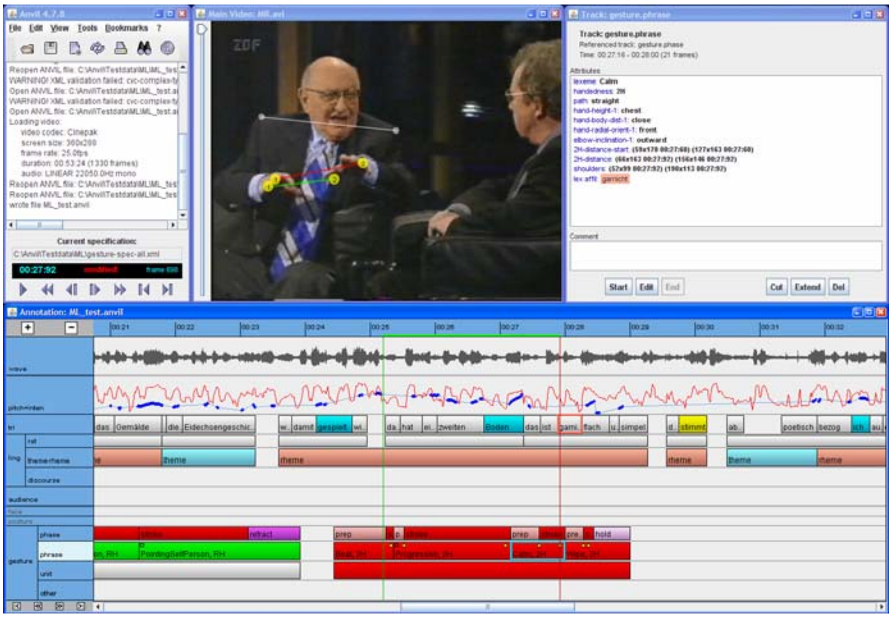
This paper presents the results of a joint effort of a group of multimodality researchers and tool developers (see [16, 17]) to improve the interoperability between several tools used for the annotation and analysis of multimodality. We propose a common exchange format for multimodal annotation, based on the annotation graph formalism, which is supported by import and export routines in the respective tools.

The paper is structured as follows: section 2 gives an overview of the multimodal annotation tools involved. Section 3 discusses the main commonalities and differences in these tools' data models and formats. Section 4 describes the main characteristics of the exchange format. Section 5 is concerned with the implementation of conversion routines from the tools' formats to the exchange format and vice versa. Section 6, finally, discusses some possible future improvements or extensions of the exchange format. The appendix contains an exemplary, commentated annotation file in the multimodal annotation exchange format.

2. Tools

The following tools were considered in this effort:

- ANVIL, a video annotation tool, developed by Michael Kipp at the DFKI in Saarbrücken (see [1, 11, 12] and figure 1). ANVIL's main application area is the study of multimodality. Other application fields include human-computer interaction, psychology, ethnology, conversation analysis and dance studies.



Reopen ANVIL file: C:\Anvil\Testdata\ML\ML_tes
WARNING! XML validation failed: cvc-complex-t
Open ANVIL file: C:\Anvil\Testdata\ML\ML_test.a
WARNING! XML validation failed: cvc-complex-t
Open ANVIL file: C:\Anvil\Testdata\ML\ML_test.a
Loading video:
video codec: Cinepak
screen size: 360x280
frame rate: 25.0fps
duration: 00:53:24 (1330 frames)
audio: LINEAR 22050.0Hz mono
Reopen ANVIL file: C:\Anvil\Testdata\ML\ML_tes
Reopen ANVIL file: C:\Anvil\Testdata\ML\ML_tes
wrote file ML_test.anvil

Current specification:
C:\Anvil\Testdata\ML\gesture-spec-all.xml
00:27:92 modified

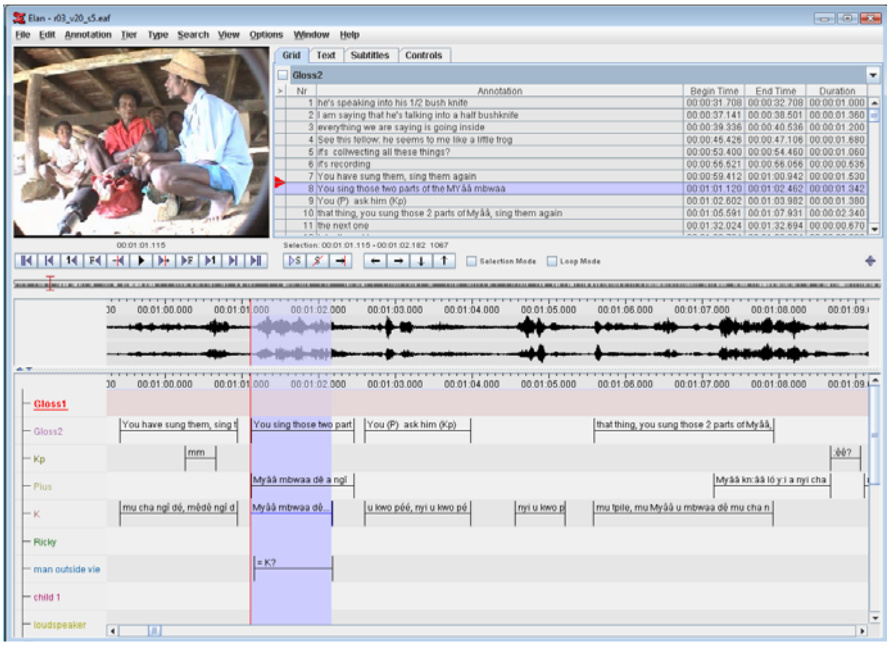
Track: gesture.phrase
Referenced track: gesture.phase
Time: 00:27:16 - 00:28:00 (21 frames)

Attributes
lexeme: Calm
handedness: 2H
path: straight
hand-height-1: chest
hand-body-dist-1: close
hand-radial-orient-1: front
elbow-inclination-1: outward
2H-distance-start: (59x178 00:27:68) (127x163 00:27:68)
2H-distance: (66x163 00:27:92) (156x146 00:27:92)
shoulders: (57x99 00:27:92) (190x113 00:27:92)
lex affil: garnicht

Comment

Fig. 1. User interface of ANVIL

- C-BAS, a tool for coding events on video or audio tracks, developed by Kevin Moffit at the University of Arizona (see [7]).
- ELAN, a tool for multi-level annotation of video and audio, developed by the Max-Planck-Institute for Psycholinguistics in Nijmegen(see [6, 9, 26]). ELAN's main application areas include studies of multimodality, studies of sign language and field linguistics (documentation of endangered languages).

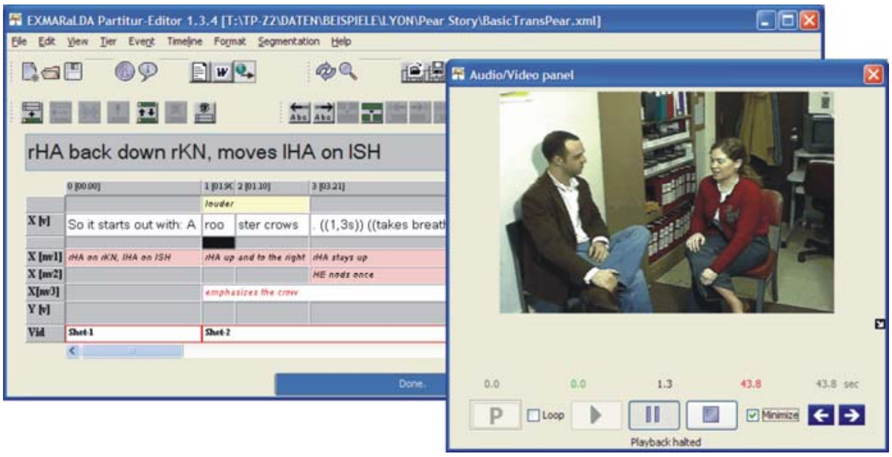


□ Gloss2

Nr	Annotation	Begin Time	End Time	Duration
1	he's speaking into his 1/2 bush knife	00:00:31.708	00:00:32.708	00:00:01.000
2	I am saying that he's talking into a half bushknife	00:00:37.141	00:00:38.501	00:00:01.360
3	everything we are saying is going inside	00:00:39.336	00:00:40.536	00:00:01.200
4	See this fellow: he seems to me like a little frog	00:00:45.426	00:00:47.106	00:00:01.680
5	it's collwecting all these things?	00:00:53.400	00:00:54.460	00:00:01.060
6	it's recording	00:00:55.521	00:00:56.056	00:00:00.535
7	You have sung them, sing them again	00:00:59.412	00:01:00.942	00:00:01.530
8	You sing those two parts of the MYåå mbwaa	00:01:01.120	00:01:02.462	00:00:01.342
9	You (P) ask him (Kp)	00:01:02.602	00:01:03.982	00:00:01.380
10	that thing, you sung those 2 parts of Myåå, sing them again	00:01:05.591	00:01:07.931	00:00:02.340
11	the next one	00:01:32.024	00:01:32.694	00:00:00.670

Fig. 2. User interface of ELAN

- EXMARaLDA Partitur-Editor, a tool for transcription of audio or video recordings of spoken language, developed by Thomas Schmidt at the University of Hamburg (see [10, 22, 23] and figure 3). EXMARaLDA is mostly used in conversation and discourse analysis, language acquisition studies and dialectology (multimodality being an issue in all fields except dialectology).
- MacVisSTA, a tool for annotation and visualization of multiple time-synchronized videos, developed by Travis Rose, Francis Queck and Chreston Miller at Virginia Tech (see [20, 21]).
- Theme, a commercial software tool for finding patterns in temporally annotated data, developed by Magnus Magnusson for Noldus (see [24]).
- Transformer, a tool for exchanging annotation data between different annotation tools and for creating customizable printable transcripts, developed by Oliver Ehmer at the University of Freiburg (see [25]).



EXMARaLDA Partitur-Editor 1.3.4 [T:\TP-Z2\DATEN\BEISPIELE\LYON\Pear Story\BasicTransPear.xml]

rHA back down rKN, moves lHA on lSH

	0 [00.00]	1 [01.9C	2 [01.10]	3 [03.21]
		louder		
X [v]	So it starts out with: A	roo	ster crows	. ((1,3s)) ((takes breat
X [nv1]	rHA on rKN, lHA on lSH	rHA up and to the right		rHA stays up
X [nv2]				HE nods once
X[nv3]		emphasizes the crow		
Y [v]				
Vid	Shot-1	Shot-2		

Audio/Video panel

0.0 0.0 1.3 43.8 43.8 sec

☑ Minimize

Playback halted

Fig. 3. User interface of the EXMARaLDA Partitur-Editor

These tools differ greatly in their technical details, in their intended target audiences, in the design of their user interfaces and in the specific tasks they help to solve. They have in common, however, that they all allow the creation of or the work with analytic textual data which is time-aligned to a video recording. They are therefore all used to carry out multi-modal annotation and analysis.

It is not uncommon, though, that a researcher wants to exchange data between two or more of them, because no single tool offers all the functionality required for a given task. For instance, a typical processing pipeline could look like this:

1) EXMARaLDA is used to transcribe the verbal behavior in an interaction,
2) ELAN's advanced video support is needed to add detailed annotation of multimodal behavior,
3) Theme is used to carry out an analysis of the annotated data,
4) Transformer is used to generate a visualization of the annotated data.

Up to a certain point, the interoperability needed for this kind of task was already provided before our effort by some of the tools in the form of import and export routines converting between the tools' own data format and that of another. However, this was an inefficient and unreliable solution because it meant that each tool developer had to keep track of and react to changes in all other tools. The obvious solution therefore was to agree on a common exchange format which can accommodate all the information contained in the individual tools' formats.

3. Comparison of data formats

As a first step towards this goal, a thorough analysis and comparison of the different tool formats was carried out. All formats have in common that their basic building blocks are annotation tuples consisting of a start and an end point (with, typically, a temporal interpretation) and one or more text labels (with no fixed interpretation). Since this is precisely the principle on which the annotation graph formalism (AG, see [4]) is based, it was natural to choose AGs as a general framework for our task. As a further advantage, several software packages facilitating the work with AGs already exist [2].

However, there are differences between the formats (1) with respect to the way the basic building blocks are organised into larger structures, (2) with respect to semantic specifications of and structural constraints on the basic and larger structural entities, and (3) with respect to organization on the storage level, for storing the underlying scheme and projects containing multiple data files. The following section discusses the differences in the formats which were identified to be relevant in terms of interoperability.

3.1. General organisation of the data structure

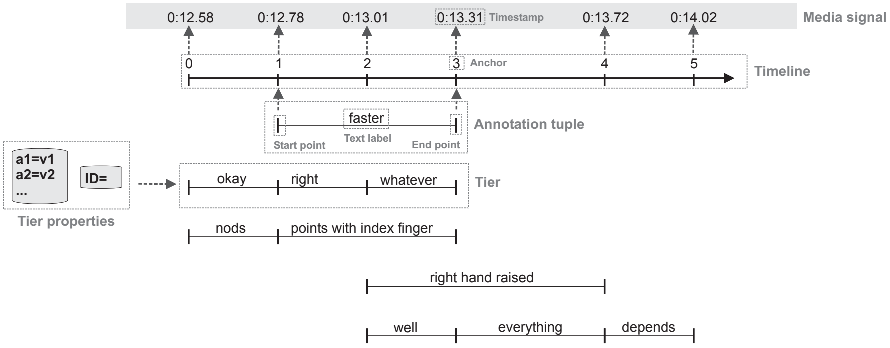


Fig. 4. Schematic diagram of basic elements of the data models

3.1.1. Tier-based data formats vs. non-tier-based formats

In ANVIL, ELAN, EXMARaLDA and Transformer, all annotations are partitioned into a number of tiers such that each annotation is part of exactly one tier, and no two annotations within a tier overlap. These tiers are usually used to group annotations which belong to one level of analysis (e.g. verbal vs. non-verbal behaviour, hand movements vs. facial expression) or to one participant (in multi-party interaction). By contrast, C-BAS, MacVisSTA and Theme do not have the concept of a tier; they keep all annotations in a single list. When converting from a non-tier-based format to a tier-based format, a partition of this list into tiers must be found. In the other direction, the grouping can be ignored or, if the target format provides this possibility, the category underlying the grouping can be represented as a property of an individual annotation.

3.1.2. Single vs. multiple labels

Annotations consist of a single label in ELAN, EXMARaLDA, MacVisSTA and Transformer, while ANVIL, C-BAS and Theme can have multiple (typed) labels for one and the same annotation. In ANVIL, these are organized by tier. Each tier T can have a set A_1-A_n of typed attributes. Every added annotation can then fill these attributes. When converting from multi-label formats to a single-label format, each multi-label annotation has to be split into a corresponding number of single-label annotations on different tiers. For instance, as figure 5 illustrates, an ANVIL track T with attributes A_1-A_n (left) would be translated to n separate tiers in ELAN (right).

```
<track name="speaker_1">
  <el start="0.12" end="0.27">
    <attribute name="words">you</attribute>
    <attribute name="POS">PRO</attribute>
  </el>
  <el start="0.27" end="0.36">
    <attribute name="words">lie</attribute>
    <attribute name="POS">V</attribute>
  </el>
</track>
```

```
<tier name="speaker_1_words">
  <annotation start="0.12" end="0.27">you</annotation>
  <annotation start="0.27" end="0.36">lie</annotation>
</tier>
<tier name="speaker_1_POS">
  <annotation start="0.12" end="0.27">PRO</annotation>
  <annotation start="0.27" end="0.36">V</annotation>
</tier>
```

Fig. 5. Correspondences between multi-label and single-label annotation

3.1.3. Implicit vs. explicit timeline

In ANVIL, C-BAS MacVisSTA, Theme and Transformer, the timestamps of annotations refer directly to media times in the recording. By contrast, ELAN and EXMARaLDA define an explicit external timeline, i.e. an ordered set of anchors to which annotations refer. Anchors in this external timeline can, but need not, be assigned an absolute timestamp which links them to the media signal (see figure 6). It is thus possible in ELAN and EXMARaLDA to leave the media offsets of certain annotations unspecified.[1]

[1] In other words: the annotator can freely determine the degree of precision of the alignment between annotations and recordings. By the same logic, it becomes possible to have a completely

```
<tier name="speaker_1_words">
  <annotation start="0.12" end="0.27">
    you
  </annotation>
  <annotation start="0.27" end="0.36">
    lie
  </annotation>
</tier>
<tier name="speaker_1_POS">
  <annotation start="0.12" end="0.27">
    PRO
  </annotation>
  <annotation start="0.27" end="0.36">
    V
  </annotation>
</tier>
```

```
<timeline>
  <anchor id="T0" time="0.12"/>
  <anchor id="T1" time="0.27"/>
  <anchor id="T2" time="0.36"/>
</timeline>


<tier name="speaker_1_words">
  <annotation start="T0" end="T1">you</annotation>
  <annotation start="T1" end="T2">lie</annotation>
</tier>
<tier name="speaker_1_POS">
  <annotation start="T0" end="T1">PRO</annotation>
  <annotation start="T1" end="T2">V</annotation>
</tier>
```

Fig. 6. Implicit and explicit timeline

In terms of interoperability, the difference between implicit and explicit timelines poses two problems: First, the former do not permit unspecified media offsets. When going from a format with an explicit to a format with an implicit timeline, missing offsets therefore have to be calculated. The simplest way to achieve this is through (linear) interpolation. Second, in going from an implicit to an explicit timeline, the question arises of how to treat points with identical offsets. If two such points are mapped to different anchors (the EXMARaLDA and ELAN formats allow this[2]), there is no straightforward way of ordering them and contradictory data structures (i.e. annotations whose endpoint precedes their startpoint in the timeline) may result. This has to be taken care of in the conversion routines.

3.2. Semantic specifications and constraints

The properties of the multimodal annotation formats discussed so far, just like the AG framework in general, are on a relatively high level of abstraction. That is, they concern very general *structural* characteristics of annotation data, and they do not say very much about their *semantics*.[3] While a high level of abstraction is

non-temporal interpretation of start and end points – for instance, when the annotated object is not a recording, but a written text.

[2] In contrast to EXMARaLDA, however, ELAN places certain restrictions on the sharing of anchors. It does not allow single anchors per time value to be shared by different annotations on tiers that are independent of each other. Sharing of anchors is only used to consistently link the begin point and end point of chains of annotations on parent and child tiers and for the construction of annotation chains on time-aligned dependent tiers that do not allow gaps. This is a kind of object-oriented approach to managing time anchors.

[3] It is in this sense that Bird/Liberman (2001:55) call their AG framework “ontologically parsimonious (if not positively miserly!)”

beneficial to interoperability in many ways, actual applications profit from more concrete semantic specifications. All of the tools considered in our effort introduce such specifications as a part of their data models and formats, thereby often imposing further structural constraints on annotations which may have to be taken into account in data conversion.

3.2.1. Speaker assignment of tiers

In EXMARaLDA and ELAN, a tier can be assigned to one member of an externally defined set of speakers. In all other tools, speaker assignment can only be expressed on the surface, e.g. by using appropriate tier names, but speakers and speaker assignment are not an integral part of the semantics of the respective data model.

3.2.2. Parent/Child relations between tiers

In Anvil, ELAN and Transformer, tiers can be explicitly assigned to a parent tier. Tied to this assignment is the constraint that annotations in child tiers must have an annotation, or a chain of annotations, in the parent tier with identical start and end points. This relationship can be used, for example, to ensure that annotating elements (e.g. POS tags) always have an annotated element (e.g. a word) they refer to. In fact, in ANVIL and ELAN, certain annotations in child tiers do not even get an immediate reference to the timeline. Instead, they inherit their start and end points from the corresponding annotation in the parent tier.[4] This increases robustness since a misalignment of such logically related elements can never occur. Apart from such structural constraint on the tiers' content, ANVIL also allows to organize tiers themselves hierarchically into groups and subgroups for visual organization (like directory folders) and inheritance of coding scheme properties (group attributes are inherited by all contained tracks).

3.2.3. Tier types

All tier-based tools define some kind of tier typology, i.e. ways of classifying individual tiers with respect to their semantic or structural properties. Thus, tiers in ANVIL can be of type ‘primary’ ‘singleton’ or ‘span’, reflecting structural properties related to the parent/child distinction described above. Similarly, ELAN dis-

[4] In ELAN, this is done through an explicit reference from an annotation in the child tier to a parent annotation on the parent tier. In certain cases, this is combined with a reference to the preceding annotation on the same (child) tier. In that way, a chain of (ordered) child annotations with the same parent is formed.

tinguishes between the ‘symbolic’ types ‘time subdivision’, ‘included in’, ‘symbolic subdivision’ and ‘symbolic association’ (Transformer uses the same distinctions), and EXMARaLDA has the tier types ‘transcription’, ‘description’ and ‘annotation’ .

All these type distinctions address a similar issue - they tell ‘their’ application about meaningful operations that can be carried out on the annotation data. However, the very fact that the typologies serve an application-specific purpose makes it difficult to map between them when it comes to data conversion.

However, some similarities exist, for instance, between ELAN-ANVIL types. Thus, the ANVIL 'singleton' type is equivalent to ELAN's 'symbolic association' (both types impose a one-to-one relationship between pairs of annotations on the two given tiers), and the ANVIL 'span' type is equivalent to ELAN's 'included in' (both types imply that an annotation on tier A (temporally) contains a series of annotations on tier B[5], possibly with gaps).

3.2.4. Restrictions on label content

Besides classifying annotations according to their structural properties, some tools also provide a way of prescribing permissible values for annotation labels. Anvil has the most far-reaching functionality in this respect – it allows the definition of possible annotation values in a separate ‘specification file’. This file stores the complete syntactic 'blueprint' of the annotation (sometimes called the coding scheme), i.e. names and types of all tiers, including their corresponding typed attributes. A similar purpose is fulfilled by a so-called ‘controlled vocabulary’ in ELAN.

There was some discussion in our group as to whether these specifications are to be considered part of the tools’ formats at all. In any case, the fact that not every tool format provides a place for specifying such restrictions on label content makes this kind of data problematic for data exchange. Moreover, the AG formalism and format do not (at least not in any straightforward way) support the definition of such sets.

[5] Similarly, EXMARaLDA requires that for every annotation A on a tier of type 'A', there must be a temporally contiguous set of annotations $B_1...B_n$ on a tier of type 'T', assigned to the same speaker, such that A and $B_1...B_n$ cover the same time interval. A is then interpreted as a (dependent) annotation of $B_1...B_n$ (e.g. a translation of a transcribed utterance). The EXMARaLDA editor, however, does not enforce these constraints during input (because EXMARaLDA was partly designed for post-editing large sets of legacy data in which such constraints had been inconsistently dealt with), but rather provides the user with a possibility to check the well-formedness of data structures in a separate step.

4. Exchange Format

Given that AG had been chosen as the general framework, we decided to develop the exchange format on the basis of AG's XML-based file format, which is identical to level 0 of the Atlas Interchange Format (AIF, see [2, 13, 19]).[6]

We agreed to use the following strategy: First, we would define the greatest common denominator of all tool formats and make sure that we achieve lossless exchange of this information. Second, we would devise a way of uniformly encoding all information which goes beyond the common denominator. In that way, the exchange format will at least capture all the available information, and each tool's import routine can decide whether and how it can make use of that information. While this manner of proceeding does not guarantee lossless round-tripping between different tools, it should at least make it possible for the user to work with a chain of tools with increasingly complex data formats without losing any information in the conversion process(es).

Essentially, the greatest common denominator consists in the basic building blocks (i.e. labels with start and end times) plus the additional structural entities (tiers and timeline) discussed in section 3.1. The concepts discussed in section 3.2., on the other hand, go beyond the common denominator information. Consequently, the main characteristics of the exchange format are as follows:

4.1. Annotations and timeline

As prescribed by AIF, annotations are represented in <Annotation> elements which refer to external <Anchor> elements via 'start' and 'end' attributes. We decided to use the 'type' feature to specify the tier. Note that in AIF there originally is no concept of 'tier'. However, it makes sense to think of different tiers as different 'types' of information (see next section). The annotation content is represented in one or more[7] <Feature> elements underneath the <Annotation> element, e.g.:

[6] It should be noted that, in one place, we did *not* adhere to the AG format specification: [19] prescribes the use of so-called 'fully qualified' identifiers for nodes, arcs, etc. in which the identifier attribute can be used to infer the place of an element in the document tree (e.g.: "an anchor 'Timit:AG1:Anchor2' belongs to the annotation graph 'Timit:AG1', which in turn belongs to the AGSet 'Timit'." [19]). Since, however, the XML specification explicitly disallows the use of colons in ID attributes (leading to various problems with XML parsers), we decided not to follow this convention in the exchange format.

[7] More than one <Feature> element is used whenever an annotation consists of more than one label (cf. section 3.1.2.). For instance, ANVIL allows a single annotation to contain a set of attributes A1..An. These can be represented in multiple <Feature name="A1">, ..., <Feature name="An"> tags.

```
<Anchor id="T6" offset="10" unit="milliseconds"/>
<Anchor id="T7" offset="30" unit="milliseconds"/> […]
<Annotation type="TIE1" start="T6" end="T7">
       <Feature name="description">And so hee</Feature>
</Annotation>
```

As mentioned above, for tools without an explicit timeline, the <Anchor> elements have to be generated from timestamps within annotations. Conversely, when a tool with an explicit timeline defines an anchor without a timestamp, the timestamp is added in the exchange format by interpolating between the nearest preceding and following timestamps that *do* have a timestamp.[8]

4.2. Tier assignment

AIF's <MetadataElement> element is used to record the existence of and information about tiers. We prescribe a fixed name 'Tier' for this kind of information and a nested <MetadataElement> element with the fixed name 'TierIdentifier' to provide each tier with a unique identifier. This identifier is then referred to from the type attribute in the <Annotation> element. Tools with non-tier-based data formats can ignore this information when importing from the exchange format, but need to generate it from appropriate other elements of the data structure (i.e. from other categorizations of annotations) when exporting to the exchange format.

```
<MetadataElement name="Tier">
   <MetadataElement name="TierIdentifier">TIE1</MetadataElement>
</MetadataElement>
 […]
<Annotation type="TIE1" start=" T6" end=" T7">
```

4.3. Additional information

Further information about tiers is stored in nested <MetadataElement> elements with the fixed name 'TierAttribute'. Each tier attribute is represented by the fixed triple Source-Name-Value, where 'Source' describes the defining instance (i.e. the tool), 'Name' the name given by the tool for that attribute and 'Value' its value.

[8] Actually AG/AIF allows anchors without timestamps, so the interpolation might just as well be deferred to the moment a tool without an explicit timeline imports the AG file. This would have the benefit that, in a conversion between tools with explicit timelines, no additional (and only approximately correct) timestamps would have to be generated. The decision not to proceed in this way is owing to pragmatic reasons (i.e. keeping conversion routines simple), but should possibly revised in future versions of the exchange format.

```
<MetadataElement name="Tier">
   […]
   <MetadataElement name="TierAttribute">
      <MetadataElement name="Source">EXMARaLDA</MetadataElement>
      <MetadataElement name="Name">speaker</MetadataElement>
      <MetadataElement name="Value">SPK0</MetadataElement>
    </MetadataElement>
</MetadataElement>
[…]
```

5. Conversion Routines

All participating tool developers were asked to write routines which would convert between their tools' formats and the exchange format. The technology for implementing these routines could be freely chosen. Thus, the ANVIL and ELAN conversions are done using the (Java) AG programming library, the EXMARaLDA conversion is based on XSL stylesheets, the Theme converter is written in Perl, the Transformer converter is written in VisualBasic, and MacVisSTA uses Python scripts for the task. At this point in time, we see no disadvantage in this diversity. Rather, we think that the fact that all these technologies have led to working conversion routines can be seen as a proof of the flexibility of our solution.

Partly, the new conversion routines have been integrated into the respective tools. Partly, they can be used as standalone converters.

6. Conclusion and Outlook

Our effort so far has resulted in a format via which the common denominator information can be reliably exchanged between the tools and which stores additional information in a standardized way. The interoperability can be extended to other tools like Praat [5], Transcriber [3] or the TASX Annotator [15] by making use of existing import and export routines which some of the tools offer. The new data exchange options and the fact that we have a systematic analysis of the tool formats' differences and commonalities are a major step forward towards the interoperability that many multimodality researchers expect from their tools.

Further developments should concern the information which goes beyond the common denominator. There are two areas in which we plan to extend the current specification of the exchange format within our approach:

- Simple partial correspondences: Some bits of information, although they do not exist in every format, are nevertheless easily mappable between those formats in which they are defined. An example is the speaker assignment of tiers which is done through a 'participant' attribute in ELAN and a 'speaker' attribute in EXMARaLDA. Mapping between these two is therefore simply a matter of agreeing that their semantics are identical and specifying a unique name for them to be used in a <MetadataElement> in AIF.

- Complex partial correspondences: Other bits of information are also present in several formats, but are encoded in non-isomorphic ways. For example, the parent-child relation between tiers is encoded in both ANVIL and ELAN as an explicit attribute which points from the child tier to the parent tier via a unique ID. In EXMARaLDA, there is no such attribute, but the relation of a tier of type 'Transcription' to all tiers of type 'Annotation' which carry the same speaker assignment is also to be interpreted as a parent-child relation. If the exchange format defines a reliable way of recording information about parent-child relations between tiers, the EXMARaLDA export could transform this information accordingly and thus make it accessible to ANVIL and ELAN for import (and vice versa). The subtle differences between ANVIL-ELAN parent/child relations have been discussed in 3.2.3. A first step towards extended exchange capabilities would be to agree on which of these relationships to include in the tier representation in the <Metadata> tags and to classify them according to properties which transcend the application level (e.g. one-to-many, one-to-one or many-to-one relationships between annotations on parent and child tier).

Going beyond our current approach, we see two ways of further enhancing tool interoperability. The first is to reduce incompatibilities by modifying and assimilating the tools' data formats themselves. However, given that the diversity in tool formats is to a great part motivated by the different specializations of the respective tools, we do not expect (nor do we think it is desirable) to fully standardize the representation of multimodal annotations in that way.

Another strategy has been proposed by a working group at the EMELD/TILR workshop 2007 at which our proposal had been presented: wherever a *format*-based approach to interoperability such as ours meets its limits, it might be worthwhile considering *process*-based methods for data exchange. In such an approach, "interoperability is achieved by having the various annotation tools interact with each other via a well-defined process which mediates the interaction among the tools. Within this process would be the requisite information regarding the data models of each tool that would interact it with as well as methods for detecting (and ideally also for resolving) annotation conflicts." (see [8]). In other words: a higher degree of interoperability could be achieved by letting a third component memorize and restore information which was lost in a conversion between two tools. Such a third component could also act on the basis of the proposed exchange format. We intend to explore this possibility in the future.

References

[1] Anvil website. http://www.anvil-software.de/

[2] ATLAS Website: http://sourceforge.net/projects/jatlas/

[3] Barras, C.; Geoffrois, E.; Wu, Z. & Liberman, M. (2000). Transcriber: Development and Use of a Tool for Assisting Speech Corpora Production. Speech Communication 33, 5-22.

[4] Bird, S. & Liberman, M. (2001). A formal framework for linguistic annotation. Speech Communication 33, 23-60.

[5] Boersma, P. & Weenik, D. (1996). PRAAT, a system for doing phonetics by computer, version 3.4. Institute of Phonetic Sciences of the University of Amsterdam, Report 132. 182 pages.

[6] Brugman, H. & Russel, A. (2004). Annotating Multimedia/Multi-modal resources with ELAN. Proceedings of LREC 2004, Fourth International Conference on Language Resources and Evaluation

[7] C-BAS website. http://www.cmi.arizona.edu/go.spy?xml=cbas.xml

[8] Cochran M.; Good, J.; Loehr, D.; Miller, S.A.; Stephens, S.; Williams, B. & Udoh, I. (2007). Report from TILR Working Group 1 : Tools interoperability and input/output formats. [http://tilr.mseag.org/wiki/index.php?title=Working_Group_1]

[9] ELAN website. http://www.lat-mpi.eu/tools/tools/elan

[10] EXMARaLDA website. http://www.exmaralda.org

[11] Kipp, M. (2001). Anvil - A generic annotation tool for multimodal dialogue. Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech). Aalborg, 1367-1370.

[12] Kipp, M. (2004). Gesture Generation by Imitation – From human behavior to computer character animation. Boca Raton, Florida: Dissertation.com.

[13] Laprun, C.; Fiscus, J.; Garofolo, J. & Pajot, S. (2002). Recent Improvements to the ATLAS Architecture. Proceedings of HLT 2002, Second International Conference on Human Language Technology, San Francisco, 2002.

[14] MacVissta website. http://sourceforge.net/projects/macvissta/

[15] Milde, J.-T. & Gut, U. (2002). The TASX Environment: An XML-Based Toolset for Time Aligned Speech Corpora. Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002), Gran Canaria.

[16] Website of the multimodal annotation workshop 2007. [http://www.multimodal-annotation.org]

[17] Rohlfing, K.; Loehr, D.; Duncan, S.; Brown, A.; Franklin, A.; Kimbara, I.; Milde, J.-T.; Parrill, F.; Rose, T.; Schmidt, T.; Sloetjes, H.; Thies, A. & Wellinghoff, S. (2006). Comparison of multimodal annotation tools: workshop report. Gesprächsforschung – Online-Zeitschrift zur verbalen Interaktion (7), 99-123.

[18] MacVissta website. http://sourceforge.net/projects/macvissta/

[19] Maeda, K.; Bird, S.; Ma, X.; Lee, H. (2002). Creating Annotation Tools with the Annotation Graph Toolkit. Proceedings of the Third International Conference on Language Resources and Evaluation, Paris: European Language Resources Association.

[20] Rose, T. (2007). MacVisSTA: A System for Multimodal Analysis of Human Communication and Interaction. Master's thesis, Virginia Tech.

[21] Rose, T. ; Quek, F. & Shi, Y. (2004). MacVisSTA: A System for Multimodal Analysis. Proceedings of the 6th International Conference on Multimodal Interfaces.

[22] Schmidt, T. (2005). Time-Based data models and the TEI guidelines for transcriptions of speech. Working papers in Multilingualism (56), Hamburg.

[23] Schmidt, T. & Wörner, K. (submitted). EXMARaLDA – Creating, analysing and sharing spoken language corpora for pragmatic research.

[24] Theme website. http://www.noldus.com/site/doc200403003

[25] Transformer website. http://www.oliverehmer.de/transformer/

[26] Wittenburg, P.; Brugman, H.; Russel, A.; Klassmann, A. & Sloetjes, H. (2006). ELAN: a Professional Framework for Multimodality Research. Proceedings of LREC 2006, Fifth International Conference on Language Resources and Evaluation.

Appendix: Commented example of an instance of the multimodal annotation exchange format

This example was generated by the EXMARaLDA export mechanism. It corresponds to the annotation file illustrated in the screenshot in figure 2.

```
<?xml version="1.0" encoding="UTF-8"?>
<AGSet xmlns="http://www.ldc.upenn.edu/atlas/ag/" xmlns:xlink="http://www.w3.org/1999/xlink"
version="1.0" id="exmaralda">

 <Metadata>
    <!-- Each tier is defined in a MetaDataElement with name 'Tier' -->
    <MetadataElement name="Tier">
       <!-- A child MetadataElement with name 'TierIdentifier' spcifies a unique ID for this tier -->
       <!-- This element is obligatory -->
       <MetadataElement name="TierIdentifier">TIE0</MetadataElement>
       <!-- Further child MetadataElements with name 'TierAttribute' -->
       <!-- define further properties of the respective tier -->
       <!-- These elements are optional -->
       <!-- this tier property says that the tier is assigned to the speaker with the (unique) ID 'SPK0' -->
       <MetadataElement name="TierAttribute">
       <!-- This MetadataElement specifies the tool which defined the property -->
             <MetadataElement name="Source">EXMARaLDA</MetadataElement>
```

```
            <!-- This MetadataElement specifies the name of the property in the tool's format -->
            <MetadataElement name="Name">speaker</MetadataElement>
            <!-- This MetadataElement specifies the value of the property -->
            <MetadataElement name="Value">SPK0</MetadataElement>
    </MetadataElement>
    <!-- Another tier property defined by EXMARaLDA: -->
    <!-- the tier is of category 'sup' (for 'suprasegmental') -->
      <MetadataElement name="TierAttribute">
            <MetadataElement name="Source">EXMARaLDA</MetadataElement>
            <MetadataElement name="Name">category</MetadataElement>
            <MetadataElement name="Value">sup</MetadataElement>
      </MetadataElement>
      <!-- Another tier property defined by EXMARaLDA: the tier is of type 'a' (for 'annotation') -->
      <MetadataElement name="TierAttribute">
            <MetadataElement name="Source">EXMARaLDA</MetadataElement>
            <MetadataElement name="Name">type</MetadataElement>
            <MetadataElement name="Value">a</MetadataElement>
      </MetadataElement>
    </MetadataElement>

    <!-- another tier definition -->
    <MetadataElement name="Tier">
            <MetadataElement name="TierIdentifier">TIE1</MetadataElement>
            <!-- follows another set of EXMARaLDA-specific tier attributes -->
    </MetadataElement>

    <!-- yet another tier definition -->
    <MetadataElement name="Tier">
            <MetadataElement name="TierIdentifier">TIE2</MetadataElement>
            <!-- follows another set of EXMARaLDA-specific tier attributes -->
    </MetadataElement>
    <!-- follow more tier definitions -->
</Metadata>

<!-- The Timeline to which Anchors refer -->
 <Timeline id="exmaralda_Timeline1">
    <!-- The Signal element specifies the media file for this annotation -->
    <Signal id="exmaralda_Timeline1_Signal1" unit="miliseconds" mimeClass=""
            mimeType="video/quicktime" encoding="" xlink:href="pear.mov"/>
</Timeline>

<!-- one AG element holds the actual Annotations and their Anchors -->
<!-- it refers to the Timeline defined above -->
<AG timeline="exmaralda_Timeline1" id="exmaralda_AG1">
    <!-- each Anchor gets a unique ID -->
    <!-- offsets are given in milliseconds -->
    <!-- Anchors should be ordered by offset -->
    <Anchor id="T0" offset="0" unit="milliseconds"/>
    <Anchor id="T1" offset="1900" unit="milliseconds"/>
    <Anchor id="T2" offset="2000" unit="milliseconds"/>
    <Anchor id="T3" offset="3211" unit="milliseconds"/>
    <Anchor id="T4" offset="5000" unit="milliseconds"/>
    <Anchor id="T5" offset="9200" unit="milliseconds"/>
    <Anchor id="T6" offset="10500" unit="milliseconds"/>
    <!-- follow more Anchor definitions -->
    <!-- each Annotation gets a unique ID -->
    <!-- the value of the 'type' attribute refers to the PCDATA value of a -->
    <!-- MetadataElement with name 'TierIdentifier' -->
    <!-- the values of the 'start' and 'end' attribute refer to the 'id' attributes of Anchor elements -->
    <!-- this Annotation element describes an annotation in tier 'TIE0', starting at anchor 'T1', -->
   <!-- ending at anchor 'T3' labelled 'louder' -->
    <Annotation id="TIE0_T1" type="TIE0" start="T1" end="T3">
            <!-- the Feature element(s) contain the actual annotation label(s)-->
            <!-- the value of the 'name' attribute can be freely chosen -->
            <Feature name="description">louder </Feature>
    </Annotation>
    <!-- [...] -->

    <!-- Three annotation elements from tier 'TIE1', describing verbal behaviour -->
    <Annotation id="TIE1_T0" type="TIE1" start="T0" end="T1">
            <Feature name="description">So it starts out with: A </Feature>
    </Annotation>
    <Annotation id="TIE1_T1" type="TIE1" start="T1" end="T2">
            <Feature name="description">roo</Feature>
    </Annotation>
    <Annotation id="TIE1_T2" type="TIE1" start="T2" end="T3">
            <Feature name="description">ster crows</Feature>
    </Annotation>
    <!-- [...] -->

    <!-- Three annotation elements from tier 'TIE2', describing non-verbal behaviour -->
    <Annotation id="TIE2_T0" type="TIE2" start="T0" end="T1">
            <Feature name="description">rHA on rKN, lHA on lSH</Feature>
    </Annotation>
    <Annotation id="TIE2_T1" type="TIE2" start="T1" end="T3">
            <Feature name="description">rHA up and to the right </Feature>
    </Annotation>
    <Annotation id="TIE2_T3" type="TIE2" start="T3" end="T4">
            <Feature name="description">rHA stays up</Feature>
    </Annotation>
    <!-- more annotations follow -->
 </AG>
</AGSet>
```