[DRAFT (19 Aug 2004) of a paper to appear in: Working papers in multilingualism (Series B). University of Hamburg: Research Centre on Multilingualism]

Time Based Data Models and the Text Encoding Initiative's Guidelines for Transcriptions of Speech

Thomas Schmidt
SFB 538 "Mehrsprachigkeit", University of Hamburg
Max Brauer-Allee 60, D-22765 Hamburg
thomas.schmidt@uni-hamburg.de

1. Motivation

The computer-readable encoding of transcriptions of spoken language[1] is a notoriously difficult area, but also one where substantial progress has been made during the last five to ten years. One of the main challenges of the task lies in the fact that spoken language, much more than written language, exhibits non-linear (parallel) relations between elements on the highest structural level, such as overlaps between speakers' utterances, the synchronicity of segmental and suprasegmental (prosodic) phenomena, and simultaneity of verbal and non-verbal behavior. Solutions developed for the computer-readable encoding of written text, particularly with regard to markup languages like SGML and XML, therefore often lead to problems when transferred to the case of spoken language.

One increasingly favored way of addressing these problems is the concept of what I will call in this paper *time based data models*. The annotation graph (AG) formalism (Bird/Liberman 2001) is arguably the most well-known exponent of this approach, but several tools and data formats currently under development build on a similar idea without necessarily claiming to be applications of AG. What AG and the other approaches have in common is that they take the temporal relation between elements as the main principle for the organization of transcription data. Irrespective of many unresolved theoretical, technological and practical issues still under discussion, this approach has already proven very useful in two respects:

- It constitutes a framework for the encoding of spoken language transcription and annotation which abstracts over differences between different transcription systems and data formats while retaining one of their substantial commonalities. In that way, it is a good candidate for facilitating the *exchange of transcription data* between different tools and computing environments and thus for aiding the reuse and archiving of costly language resources.
- It can serve as the basis for the *construction of flexible while user-friendly software tools* for the transcription and annotation of spoken language. The flexibility of such tools is a consequence of the degree of abstraction of the data model: since it solely relies on the temporal ordering of elements which is hardly a matter of linguistically or otherwise theoretically motivated debate, it can be used with a variety of different transcription systems devised by and for researchers from different theoretical backgrounds. The user-friendliness of such tools, on the other hand, arises from the fact that the data model is simple and intuitive to comprehend and – more importantly – can be *visualized* in a simple

[1] I prefer to speak not of 'speech', but of 'spoken language' in this context because the term 'speech' runs the risk of being tightly associated with the 'speech technology' community whereas the people interested in the kind of work presented here are more likely to be found among linguists, conversation analysts and so forth. For a similarly motivated distinction, see also Leech (1995).

and intuitive manner in the form of two-dimensionally organized notational forms like musical score notation or column notation (see below for an elaboration of this point).

The Text Encoding Initiative's approach to the encoding of transcriptions of spoken language, on the other hand, is not a time based one – it is hierarchy based. This means that the principle relation between any two elements in a TEI document is not defined by their respective positions on a timeline, but by their positions in an ordered hierarchy. Parallel relations thus become an exception to the rule and have to be encoded by means not provided by the top-level structural organization of the data. Furthermore, in contrast to most time-based data models, the TEI guidelines do not entirely set aside ontological specifications[2], i.e. they make explicit assumptions about what elements may typically occur in a transcription of spoken language (e.g. utterances, pauses, etc.) and formulate suggestions (i.e. they provide tag sets) for handling these elements in a uniform manner.

So, at first glance, time based data models and the TEI guidelines for transcriptions of speech are quite dissimilar concepts. However, since they address quite similar needs, namely the enhancement of computer processability and exchangeability of transcription data, it seems desirable to find ways of bringing the two together. This appears all the more promising given that a second glance reveals that both concepts already comprise some constructs that may serve as a first step towards a compatibility between time based, ontologically empty and hierarchy based, ontologically specified data models; in particular:

- The TEI guidelines for transcriptions of speech suggest the concept of a timeline for expressing temporal relationships that are not covered by the hierarchical structure of the document.
- Conversely, time based data models, though not hierarchy based on a conceptual level, are typically encoded on the physical level[3] as XML files and in that way always confronted with issues of compatibility between time based and hierarchy based conceptions of data.
- Although the TEI guidelines make precise suggestions for quite a number of specific elements that may occur in a transcription of spoken language, they neither require that all of these elements be used nor that a description be limited to these elements. Rather, the guidelines acknowledge that the set of elements necessary for any particular research or documentation purpose cannot be foreseen in its entirety. Therefore, they contain some constructs that enable their users to supplement the predefined tag sets if required.
- Conversely, many implementations of time based data models are not totally ontologically empty – they contain at least some kind of distinction of arc or tier types[4] on the basis of which certain processing steps are made possible (see further down for an elaboration of this point).

Hence, the aim of this paper is to explore how time based data models and the TEI guidelines for transcriptions of speech fit together. The benefits of an answer to this question should be obvious: On the one hand, it would allow users to use a variety of existing tools to create TEI-

[2] Bird/Liberman (2001: 55), for instance, are very clear about the AG formalism's abstinence with respect to ontological specifications: "We have tried to demonstrate generality, and to provide an adequate formal foundation, which is also ontologically parsimonious (if not positively miserly!)."

[3] I refer here to a three-level-architecture of data processing as described, for instance, in Date (1995: 28f). Bird/Liberman (2001: 25) very clearly state that their concept is also based on such an architecture, whereas the TEI guidelines do not explicitly say how their markup based approach relates to these three levels.

[4] In that way they meet an expectation formulated by Bird/Liberman (2001: 25) for the AG formalism: "The formalization presented here is targeted at the most abstract level: we want to get the annotation formalism right. We assume that system implementations will add all kinds of special-case data types (i.e. types of labels with specialized syntax and semantics)."

conformant data[5]. On the other hand, it would place time based data models, which are primarily designed for the description of *spoken* language data, into the broader context of a widely known and used standard, which is also (even chiefly) concerned with the description of many types of *written* language data.

There is certainly more than one way of bringing the two data models together, and I do not claim to cover (or even have an idea of) them all, nor do I think that the concept developed in the following sections is in any way superior to other solutions that may arise. Rather, my aim is to formulate as concretely as possible one scenario where one particular time based data model is brought into accordance with one particular subset of the TEI guidelines for transcriptions of speech. In order to make clear the usefulness of this solution, I will put a special emphasis on aspects of application.

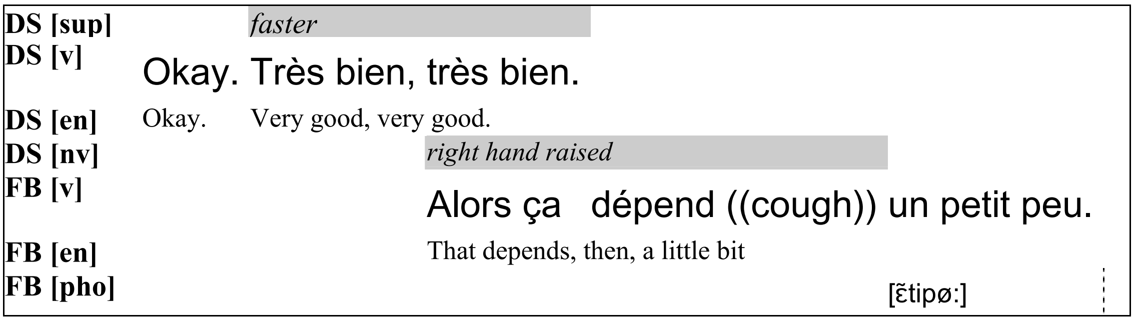
2. The "single timeline, multiple tiers" data model

I will start by describing a very simple time based data model, variants of which are used as the basis for the data formats of at least five transcription tools currently under development:

- the ANVIL tool, developed at the University of Saarbrücken (Kipp 2001),
- the EUDICO Linguistic Annotator (ELAN), developed at the Max-Planck-Institute in Nijmegen (Brugman 2004),
- the EXMARaLDA Partitur-Editor developed at the University of Hamburg (Schmidt 2004a),
- the Praat software, developed at the University of Amsterdam (Boersma/Weenik 1996) and
- the TASX Annotator developed at the University of Bielefeld (Milde/Gut 2002).

In this section, I will abstract over differences between these variants and describe the common underlying concept under the notion of "single timeline, multiple tiers data model", abbreviated *STMT*.

Consider the following excerpt of a transcript in musical score notation (cf. Ehlich 1992):



DS [sup]		*faster*			
DS [v]	Okay.	Très bien,	très bien.		
DS [en]	Okay.	Very good,	very good.		
DS [nv]			*right hand raised*		
FB [v]			Alors ça	dépend ((cough))	un petit peu.
FB [en]			That depends, then, a little bit		
FB [pho]					[ɛ̃tipø:]

Figure 1: Transcript example in muscial score notation

This excerpt exemplifies many of the characteristics of a transcription of spoken language, namely:

- It represents (in orthographic transcription) the words uttered by the participating speakers (*DS* and *FB*) in their temporal sequence (reading from left to right in the tiers titled *DS [v]* and *FB [v]*).

[5] At present, there is, to my knowledge, no sophisticated transcription tool operating on TEI data. Of course, standard XML editors will facilitate the input of TEI transcriptions in some way, but, in my experience, this support alone will not be viewed as adequate by transcribers.

- It subdivides connected sequences of words into smaller units (the boundaries of these units appear in the form of punctuation – a comma and three periods).
- It represents temporal overlap between certain words of different speakers (reading from top to bottom, the second '*très bien*' of speaker DS overlaps with the '*Alors ça*' of speaker FB).
- It represents, in their precise temporal extension, certain prosodic features of the words uttered (speaker DS speaks *faster* while uttering the words '*Très bien, très bien*').
- It represents, in their precise temporal extension, certain non-verbal phenomena that interrupt the stream of speech of the speakers (between the words '*dépend*' and '*un*', speaker FB *cough*s)
- It represents, in their precise temporal extension, certain non-verbal activities that accompany the stream of speech of the speakers (speaker DS has his *right hand raised* starting when he utters the second '*très bien*' end ending after speaker FB *cough*s)
- Beside these elements describing actual verbal or non-verbal behavior of the participants, the example also contains additional analytic pieces of information that refer to other elements of the transcription rather than directly to the transcribed recording (the translation of the speakers' utterances into English and a precise phonetic transliteration of the words '*un petit peu*' by speaker FB).

An intuitive and simple conception for an underlying data model is illustrated in figure 2:

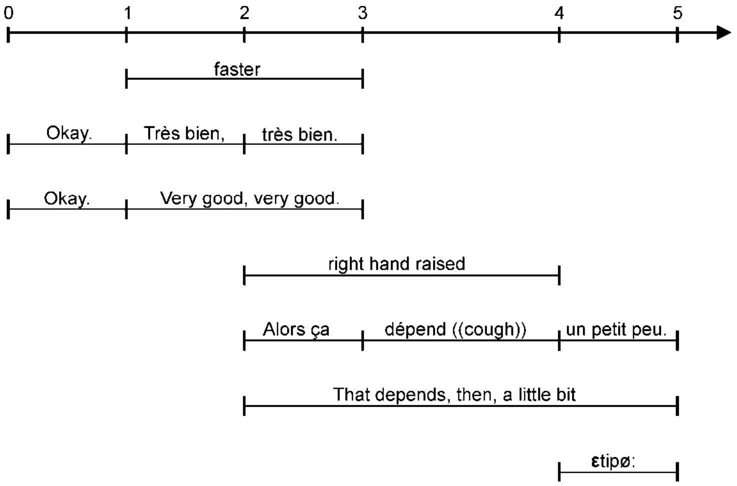


Figure 2: ‚Single timeline, multiple tiers' data model

It consists of a *timeline*, subdivided into 5 discrete intervals by 6 *time points* (numbered from 0 to 5), and of 12 *event descriptions*, anchored to this timeline via a *start point* and an *end point*, and distributed over 7 *tiers* such that, within any single tier, any two event descriptions will not overlap. Since the timeline is fully ordered – the temporal relation of any two time points can always be determined –, and every event description only refers to points of that single timeline, event descriptions, when viewed as atomic units, are also fully ordered, i.e. the temporal relation between any two event descriptions can always be determined[6]. Note that the order in the timeline is a purely relative one: it does not depend on absolute time val-

[6] Possible temporal relations are

- sequence (the time intervals of events A and B follow one another),
- partial overlap (the time intervals of events A and B share a common part),
- total overlap (the time interval of event A comprises the time interval of event B or vice versa),

with further subdistinctions. For a classification of such relationships, see also Sasaki/Witt (2004: 656).

ues referring to an underlying media signal. However, it is of course possible to assign absolute time values to some or all of the points in the timeline without altering the principle organization of the data structure.

Simple as this data model may be, it already supports a number of useful processing steps in the work with transcription data. Concerning the *input* of transcription data, it is a conveniently easy and intuitive concept for a user interface which allows the transcriber to select portions of a digitized media signal (e.g. a sound file) and enter descriptions of these portions on different linguistic levels[7]. The following screen shot taken from Praat illustrates this:

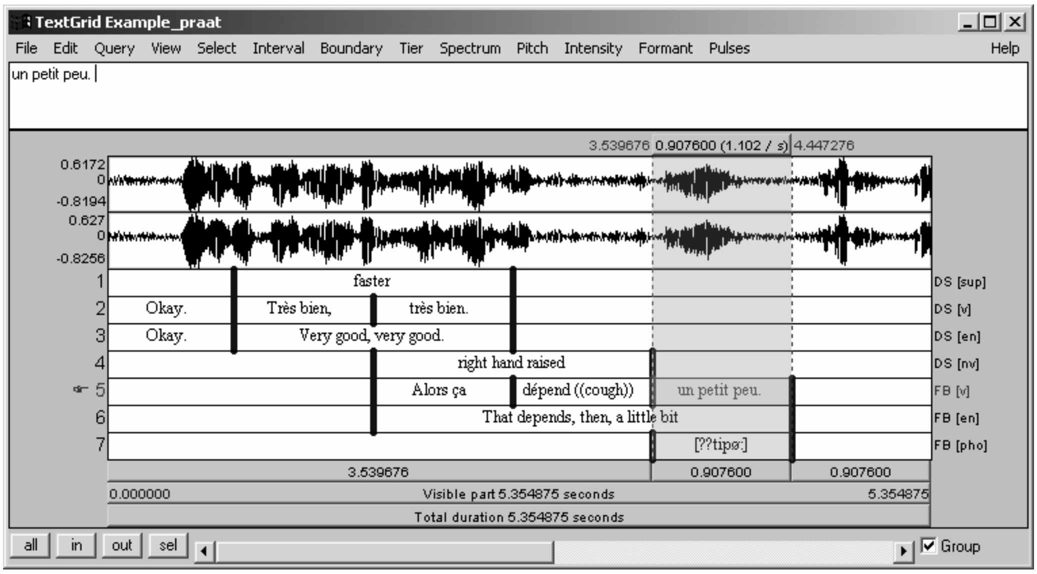

TextGrid Example_praat
File Edit Query View Select Interval Boundary Tier Spectrum Pitch Intensity Formant Pulses Help
un petit peu.
3.539676 0.907600 (1.102 / s) 4.447276
0.6172
0
-0.8194
0.627
0
-0.8256
1 faster DS [sup]
2 Okay. Très bien, très bien. DS [v]
3 Okay. Very good, very good. DS [en]
4 right hand raised DS [nv]
5 Alors ça dépend ((cough)) un petit peu. FB [v]
6 That depends, then, a little bit FB [en]
7 [??tipø:] FB [pho]
3.539676 0.907600 0.907600
0.000000 Visible part 5.354875 seconds 5.354875
Total duration 5.354875 seconds
all in out sel
Group

Figure 3: Screenshot of Praat

Concerning the *output* of transcription data on screen or paper, the information encoded in the STMT data model is sufficient at least to calculate two classical types of visualization: an example of a visualization following the layout principle of *musical score (or partitur) notation* has already been given in figure 1 above. This type of output is predominantly used in discourse and conversational analysis and has the advantage of accommodating established reading habits (left-to-right reading direction) while at the same time allowing the representation of simultaneity in a manner that is familiar from musical notation. A visualization following the layout principle of *column notation* is given in figure 4 below. This type of output is mainly employed in child language acquisition studies because it is well suited to emphasize the asynchronous nature of parent-child-interaction (see Ochs 1979).

	DS [sup]	DS [v]	DS [en]	DS [nv]	FB [v]	FB [en]	FB [pho]
0		Okay.	Okay.				
1	faster	Très bien,	Very good, very good.				
2		très bien.		right hand raised	Alors ça	That depends, then, a little bit	
3					dépend ((cough))		
4					un petit peu.		[ɛ̃tipø:]

Figure 4: Transcript example in muscial score notation

[7] And it remains an suitable concept for an intuitive user interface also when there is no digitized signal available from which to navigate the transcription process. See the screenshot of the EXMARaLDA Partitur-Editor below.

Concerning computer-assisted *analysis* of transcription data (like querying or semi or fully automatic annotation), however, this most simplified version of the STMT data model quickly reveals its limitations: since all information is indifferently represented in event descriptions, the data model abstracts over possibly vital differences of information types; and since all structural information is based on the assignment of these description units to a single, fully ordered timeline, some possibly vital structural relations may not be representable. The concepts presented in the two following sections are in essence attempts to overcome these limitations.

3. The EXMARaLDA Basic-Transcription

A Basic-Transcription is one of three XML file formats used in the EXMARaLDA system. The data model underlying this file format is a STMT model with some extensions for handling transcription meta-data (information about recordings, transcribers, speakers etc.) and some extensions for handling additional distinctions between event descriptions. I will concentrate on the latter in this section.

A distinction between different types of event descriptions is made on the level of tiers. The Basic-Transcription data model allows an assignments of tiers to a set of *categories* and to a set of *speakers*[8]. In that way, it becomes possible to express fundamental differences and commonalities between event descriptions, for instance that – in the above example – the actions described in the events of tier 2 and 5 are all *verbal* actions and that the phenomena described in the events of tier 1, 2, 3 and 4 all 'belong' to the same *speaker*. A further refinement of the description is attained by attributing each category to one of three pre-defined *types*.

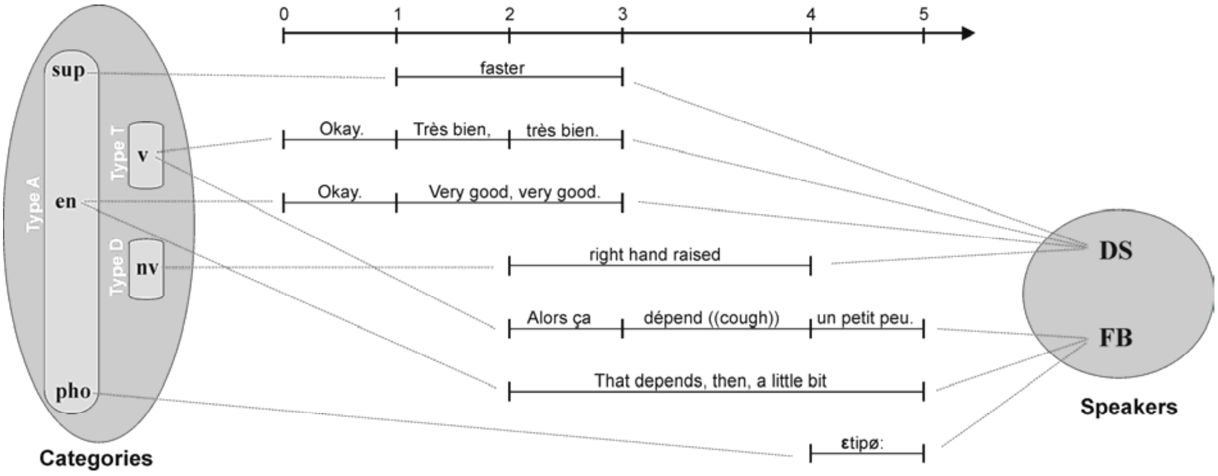


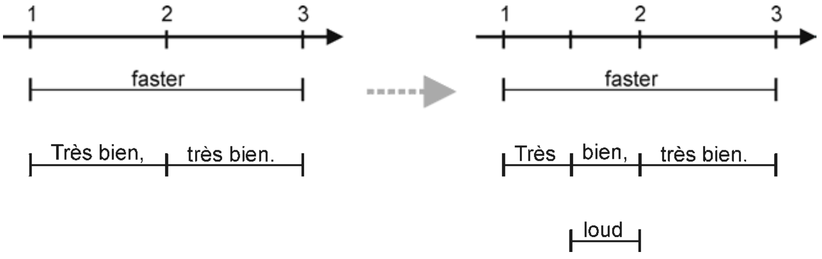
Figure 5: The EXMARaLDA Basic-Transcription data model

These pre-defined types are **T**(ranscription), **D**(escription) and **A**(nnotation).
The distinction between event descriptions of type **A** on the one hand and event descriptions of type **T** and **D** on the other hand is basically the same as the distinction between weak and regular entities in database theory[9]: whereas transcriptions and descriptions get their assign-

[8] In other variants of the STMT tiers model, for instance those used by Praat and by the TASX Annotator, it is only possible to assign a tier to a set of labels making it difficult to distinguish between type and speaker of an event. The ELAN data model is similar to EXMARaLDA in this respect.

[9] Date (1995: 351ff), for instance, gives the following definition: "A weak entity is an entity that is existence-dependent on some other entity, in the sense that it cannot exist if that other entity does not also exist. […] A regular entity, by contrast, is an entity that is not weak."

ment to the timeline independently of other entities (their reference to the timeline is immediate), annotations only have an indirect relation to the timeline – their primary structural feature is not their temporal extent, but the fact that they specify a property of another transcribed entity. Thus, a sensible restriction on the structure of a Basic-Transcription is to require that for every description *X* of type **A** of a given speaker, there has to be an event description or a sequence of event descriptions of type **T** of the same speaker that shares its start end and point with *X*. In the practice of computer-assisted corpus analysis, this restriction will, for instance, allow a query mechanism to look for a certain feature (e.g. '*faster'* in the example) and then output all the instances of event descriptions that this feature belongs to (i.e. the chain of words '*Très bien, très bien.*' in the example).
The distinction between event descriptions of type **T** and event descriptions of type **D**, on the other hand, is a distinction between types of symbolic description. Tiers of type **D** contain only *atomic* descriptions, that is strings of symbols that can neither be subdivided into smaller meaningful units nor be combined to larger meaningful units. In the example, the description '*right hand raised*', for instance, does *not* describe an event consisting of two subsequent events with the descriptions '*right hand*' and '*raised*'[10]. This is different for events in tiers of type **T**. The concept of horizontally aligning simultaneous events in musical score notation, and, in fact, the whole concept of transcription itself, relies heavily on the fact that sequences of entities of written language, like entities of spoken language can be meaningfully segmented and combined[11]. Thus, in the example, if one wanted to add a further annotation stating that the first '*bien*' uttered by speaker DS is *loud* in comparison with the rest of his speech, one could add a suitable point to the timeline and segment the description '*Très bien*' into two descriptions '*Très*' and '*bien*' the second of which would then be the reference event for the annotation:

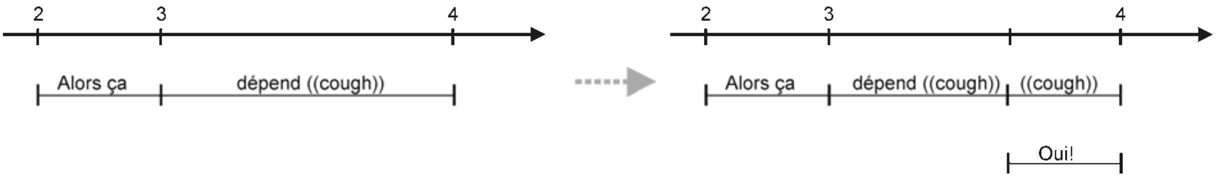


Conversely, this property of events in tier of type **T** makes it possible to combine event descriptions to larger units. This can be particularly useful for deriving a more hierarchized representation of the elements of a STMT data set and is thus a prerequisite for transforming STMT data into TEI data (see the following sections for an elaboration of this point).

However, the example also illustrates two kinds of exceptions to the rule of segmentability and combinability of event descriptions in tiers of type **T**.
One exception is the description of speaker FB's *cough* that is inserted between the description of the words uttered. This section of the symbolic transcription cannot, like the rest, be

[10] If one wanted to subdivide this event into two events, one would either have to repeat the whole description for each of them or choose two completely different symbolic descriptions (like '*raises right hand*' and '*lowers right hand*') that cannot be formally derived from the original description.
[11] From the point of view of a symbol manipulator like the computer, this is the essence of Martinet's (1960) concept of "double articulation". For people familiar with markup languages, this may seem like a trivial observation, because markup languages, in their distinction between information encoded in character data (segmentable) and information encoded in tag names and attributes (atomic), support this feature very transparently. The AG framework as proposed by Bird/Liberman (2001), however, does not pay attention to this fundamental distinction between atomic and non-atomic symbolic descriptions and thus neglects one of the salient characteristics of language description.

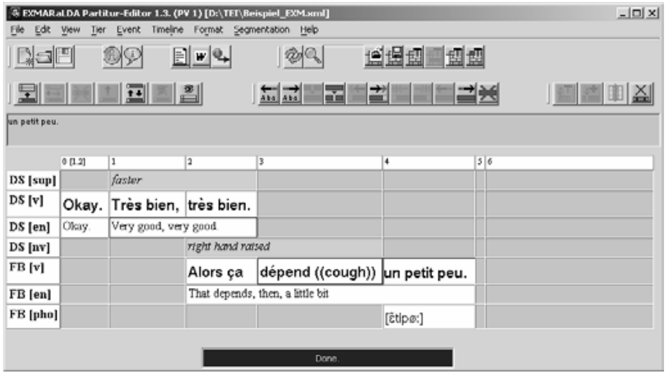
meaningfully subdivided into smaller parts. This is not only problematic in terms of readability, it could also become relevant if, for instance, the transcriber wanted to add a description of an event that partially overlaps the coughing (e.g. a third speaker uttering the word '*Oui*'). Most transcription systems have developed solutions for these kinds of problem. In the example, the string '*cough*' is visually separated from the rest of the transcription by a pair of double round brackets[12]. This visual clue can also serve as a kind of implicit markup signaling to a computer application that this part of the description has to be treated differently from the preceding and following context. Concerning the non-segmentability of such descriptions, the prevalent solution is to split the event in two events both of which are assigned the original description:



The other exception is the use of punctuation. Unlike the sequences of graphemes forming a representation of the words uttered, the spaces between words, the comma and the periods do not integrate themselves into a logic of temporal sequence paralleling the sequence of events in the transcribed interaction. Rather, these punctuation elements serve to mark the end points of linguistic units – spaces occur at the end of words, the periods terminate utterances and the comma marks the end of the first part of a repetition[13]. The punctuation elements thus also constitute a kind of implicit markup which, when applied consistently and unambiguously, can serve as the basis of an automatic segmentation of these strings by a computer program.

To summarize, the assignment of tiers to speakers and categories and the classification of categories into three pre-defined types are the main specifications that an EXMARaLDA Basic-Transcription adds to the general STMT data model described in the previous section.

Basic-Transcriptions are stored in XML files (as exemplified in Appendix A) and are used as the storage format for the EXMARaLDA Partitur-Editor, a tool for input and output of transcriptions in musical score notation. As this data model is very similar (though not identical) to those used by ELAN, Praat and the TASX annotator (see above), the Partitur-Editor can also provide export and import filters for a data exchange with these tools.


EXMARaLDA Partitur-Editor 1.3. (PV 1) [D:\TEI\Beispiel_EXM.xml]
File Edit View Tier Event Timeline Format Segmentation Help
Done.

un petit peu.

	0 [1.2]	1	2	3	4	5	6
DS [sup]		*faster*					
DS [v]	Okay.	Très bien,	très bien.				
DS [en]	Okay.	Very good, very good					
DS [nv]			*right hand raised*				
FB [v]			Alors ça	dépend ((cough))	un petit peu.		
FB [en]			That depends, then, a little bit				
FB [pho]					[ɛ̃tipø:]		

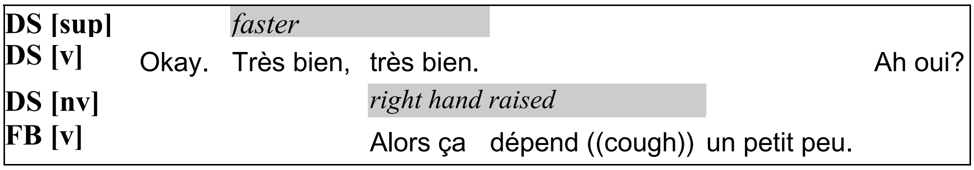
[12] The example follows the HIAT transcription convention (Ehlich 1992, Rehbein et al. 2004). Other conventions use a different type of bracketing or capitalization for the same purpose.

[13] Again, this only holds for transcriptions following the HIAT convention. What punctuation marks are used and what they actually mean differs from transcription system to transcription system. In the tradition of conversation analysis, for instance, periods as well as commas mark the end of intonation units and, at the same time, characterize the intonation movement on that unit.

4. Beyond the single timeline

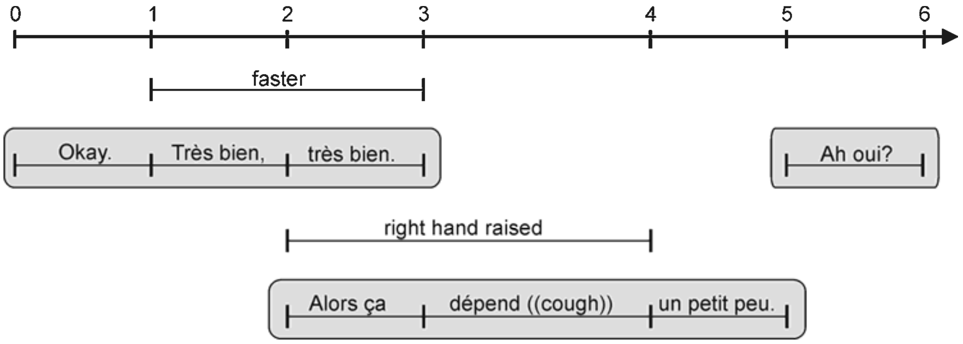
The STMT data model allows the transcriber to express the *temporal* structure of an interaction in a reasonably precise way. Whenever a new event occurs, he can add convenient points to the timeline such that the start and end points of that event can be related to the start and end points of other events taking place around the same time. However, this temporal subdivision is bound to be irregular with respect to *linguistic* units – a speaker may start his turn in the middle of another speaker's word, gestures and mimics may accompany speech without a uniform relation to the turns or words uttered, and the need to represent these temporal relations as accurately as possible will force the transcriber to distribute the description of a turn over several events or to interrupt the description of a word by an event boundary. One cannot therefore assume that event boundaries in a STMT data set will always coincide with the boundaries of meaningful linguistic units (i.e. that each event will constitute one and only one linguistic unit). As, however, being able to identify the boundaries of linguistic units is an indispensable prerequisite for many computer-assisted processing steps (e.g. for POS tagging or querying), the STMT data model needs to be extended by a possibility to supplement the representation of the *temporal* structure of speech events by a representation of their *linguistic* structure.

A first step towards such an extension is the concept of a *segment chain*. A segment chain is defined as a maximally long uninterrupted sequence of events in a tier of type T. In order to illustrate this, we add a second utterance by speaker DS to the above example (and leave out some annotations):



DS [sup]		*faster*				
DS [v]	Okay.	Très bien,	très bien.			Ah oui?
DS [nv]			*right hand raised*			
FB [v]			Alors ça	dépend ((cough))	un petit peu.	

There are two tiers of type T in this transcription (the second and the fourth), and these contain altogether three segment chains:



Since events in tiers of type T are combinable (see above), these segment chains naturally lend themselves to a hierarchical XML representation of the following kind, where the start and end point of the superordinate element can be derived from the start and end points of the subordinate elements.

```
<segment-chain start="T0" end="T3">
    <event start="T0" end="T1">Okay. </event>
    <event start="T1" end="T2">Très bien, </event>
    <event start="T2" end="T3">très bien. </event>
</segment-chain>
```

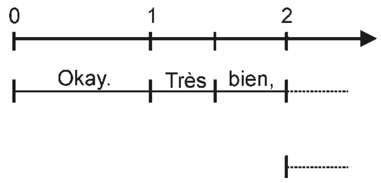
This additional structuring of events into segment chains already has a useful application – it can serve to calculate a visualization of the transcription in a line for line notation[14]:

DS: Okay. Très bien, [très bien.]
FB: [Alors ça] dépend ((cough)) un petit peu.
DS: Ah oui?

Moreover, segment chains constitute a convenient starting point for a segmentation of events into smaller linguistic units. Since they group consecutive events into a larger entity which is maximal by its definition, it is reasonable to assume that all other linguistic units – be they words, phrases, intonation units or others – can be hierarchically subordinated to them, i.e. every linguistic unit will be part of one and not more than one segment chain. For instance, the first segment chain in the above example could be segmented into utterances[15] and words in the following way (we ignore the punctuation for the time being):

```
<segment-chain>
    <utterance>
        <word>Okay</word>
    </utterance>
    <utterance>
        <word>Très</word>
        <word>bien</word>
        <word>très</word>
        <word>bien</word>
    </utterance>
</segment-chain>
```

However, such a segmentation reveals a further limitation of STMT data model. In order to integrate this segmentation into the logic of this model, each word would have to be assigned a start and an end point in a fully ordered timeline. This is unproblematic as long as there are no other timepoints competing with those necessary to mark the word boundaries. The first '*Très bien*' of speaker DS, for instance, can be segmented into words with the help of an additional point on the timeline between points 1 and 2:

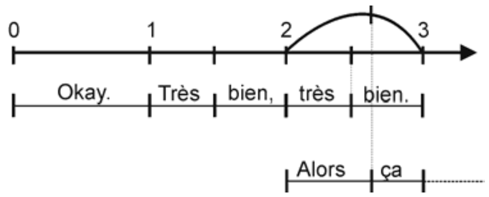


[14] This type of notation is the third classical notational form besides musical score and column notation. Note that without the grouping of events into segment chains, only a line for line notation of the following kind would be possible which is much harder to read because it spreads coherent streams of words over several lines:

DS: Okay.
DS: Très bien,
DS: [très bien.]
FB: [Alors ça]
FB: dépend ((cough))
FB: un petit peu.
DS: Ah oui ?

[15] The term *utterance* here is to be read, once more, in the context of the HIAT convention. Other transcription systems have a different notion of this term or do not use it at all, but instead provide a different unit for a subdivision of segment chains above the word level (e.g. the intonation unit in conversation analytic transcription systems).

For the second ‘*très bien*’, however, there is a conflict between the timepoint necessary to segment these two words and the timepoint necessary to segment the ‘*Alors ça*’ uttered simultaneously by speaker FB. In order to conform to the STMT data model which requires the timeline to be fully ordered, a transcriber would have to determine the exact temporal relation between the starting points of the words ‘*bien*’ and ‘*ça*’. While this of course possible in theory (the two words do have an objective temporal relationship to one another), it may prove unfeasible in practice – in cases of overlap, it is often difficult enough to identify the start and end points of the entire overlapping stretch, and the quality of the recording (as well as considerations of time and money spent for the transcription process) may make it seem unreasonable to aim at an even higher degree of precision[16]. In order to be able to integrate such segmentations into a time based data model, one therefore has to loosen the restrictions of the STMT data model. One way to do this is to allow *bifurcations of the timeline*, i.e. sections between two fully ordered timepoints in which there is no definite temporal relationship between points belonging to different *timeline forks*.



In that way, the principle metaphor of time based data models – every entity must refer to a timeline – is retained, but modified in a way that allows the transcriber to encode the possibly conflicting temporal and linguistic structure of a spoken language interaction in one and the same data set.

5. The EXMARaLDA Segmented-Transcription

A Segmented-Transcription is another of the three XML file formats used in the EXMARaLDA system. The data model underlying this file format is that of a Basic-Transcription extended by the possibilities of combining and segmenting events into linguistic units with the help of a bifurcated timeline as elaborated in the previous section.

5.1. Timed Segments, Atomic Timed Segments and Non-Timed Segments

The EXMARaLDA system is intended as a framework for computer-assisted transcription and annotation that is independent of a particular transcription system. As Ochs (1979) has demonstrated, “transcription is a selective process, reflecting theoretical goals and definitions” and hence every transcription system will necessarily define its “own” set of entities of spoken language. For instance, whereas system A may provide the concept of an *utterance* to divide the stream of speech into smaller units, system B may use the (theoretically different) concept of an *intonation unit* for the same purpose. Similarly, the concept of *word*$_1$ in system A need not necessarily match the concept of *word*$_2$ in system B. The considerations about segmentation in the previous section are therefore dependant on the definitions of an underlying transcription system. In order to reflect this dependency and the diversity in segmentation entities that may result from it, the EXMARaLDA system does not provide a pre-defined set of units for the subdivision of segment chains. Instead, it follows the approach of tier types and categories (see section 3) in that it allows the user to freely assign a *name* to each seg-

[16] In fact, most transcription systems devised for conversation and discourse analysis or for language acquistion studies state explicitly that the transcriber only has to determine the start and end point of an overlap and can ignore the temporal relation of entities within simultaneous stretches.

ment, but also to differentiate between different *types* of segments according to their formal properties. The possible types are timed segments (**TS**), non-timed segments (**NTS**) and atomic timed segments (**ATS**).

Timed segments are those whose symbolic descriptions can be segmented and combined in the way described in section 3. Thus, only a timed segment can contain other segments. The description of words are of this type as well as the descriptions of utterances and of entire segment chains.

Non-timed segments are segments that cannot be integrated into a logic of temporal sequence. As described in section 3, the punctuation marks fall under this type.

The first segment chain in the example could thus be represented as an ordered hierarchy of timed and non-timed segments in the following way[17]:

TS segment chain										
TS utterance		**TS** utterance								
TS w	**NTS** p	**TS** w	**NTS** p	**TS** w	**NTS** p	**NTS** p	**TS** w	**NTS** p	**TS** w	**NTS** p
Okay	.	*Très*		*bien*	,		*très*		*bien*	.

Atomic timed segments, finally, are segments that can be integrated into a logic of temporal sequence, but whose symbolic descriptions cannot be further subdivided. Descriptions of non-verbal events that interrupt the stream of speech (like the *cough* in the example) are of this type. The second segment chain in the example could thus be represented as an ordered hierarchy of timed, non-timed and atomic timed segments in the following way:

TS segment chain													
TS utterance													
TS w	**NTS** p	**TS** w	**NTS** p	**TS** w	**NTS** p	**ATS** non-pho	**NTS** p	**TS** w	**NTS** p	**TS** w	**NTS** p	**TS** w	**NTS** p
Alors		*ça*		*dépend*	((	*cough*	))	*un*		*petit*		*peu*	.

According to their formal properties, only timed segments and atomic timed segments are assigned start and end points on a (possibly bifurcating) timeline.

The hierarchic organization of this data structure (larger timed segmented dominate smaller timed segments, non-timed segments and atomic timed-segments) can be exploited for the XML encoding of a Segmented-Transcription as exemplified in appendix B: the temporal (event) structure and the linguistic (segment) structure are encoded in separate segmentation elements that are interrelated by their reference to the timeline and by the top-level organization of tiers into segment chains – segment chains in different segmentations that share start and end points will be identical with respect to character data, but different with respect to the intervening elements[18].

[17] 'w' (word), 'p' (punctuation), 'utterance', 'segment chain' and – in the next figure – 'non-pho' (non-phonological phenomenon) are the names of these segments. They are not pre-defined by the system, but can be chosen in accordance with a given transcription system. Again, the system underlying this example is HIAT.

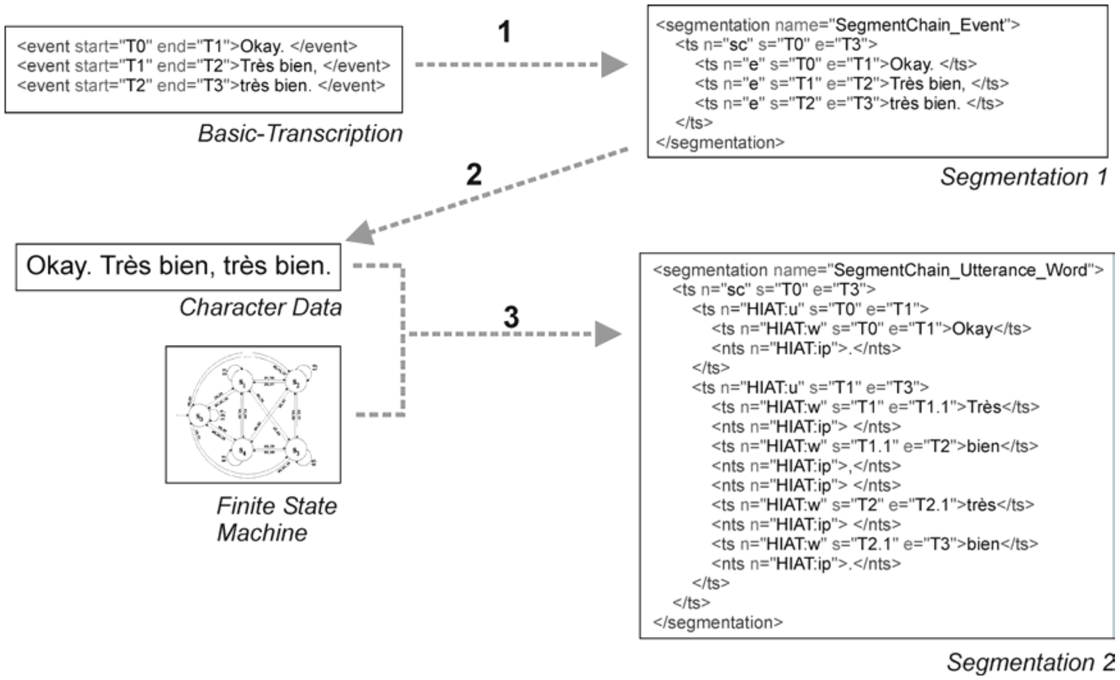
[18] On the level of segment chains in one tier, this approach is similar to the one presented in Sasaki/Witt (2004: 655): "[…] we annotate the same textual resource several times. This annotation technique results in a set of annotated XML instances differing only in the markup, i.e. the elements, attributes and attribute values. Because the textual content of all layers is identical, the text can serve as a link between these layers."

5.2. Automatic Segmentation by Finite State Machines[19]

Although a Segmented-Transcription still has a reasonably straightforward structure in so far as it uniformly retains the temporal ordering of elements as the top-level structural relation, it is certainly too complex to be read or written as a plain XML file by a human. Moreover, the bifurcating timeline and the interrelatedness of segment chains in different segmentations also make it difficult to construct a software tool that would allow an efficient interactive editing of these structures in a graphical user interface. In the EXMARaLDA system, Segmented-Transcriptions are therefore not directly created by the user, but automatically generated from Basic-Transcriptions. This manner of proceeding builds on two assumptions:

1) There must be a point in the transcription workflow where a transcriber can say that a Basic-Transcription is *completed*, i.e. where no further changes to the temporal structuring and the symbolic description of events are to be expected.
2) The conventions used in describing the events must be *formalized* to such a degree that the implicit markup of segment boundaries (like spaces separating words or brackets marking the insertion of a non-phonological element) can be used as a reliable indicator for a calculation of an explicit segmentation.

If these two conditions are met, a segmentation algorithm can be devised that takes a completed Basic-Transcription as its input and, on the basis of the regularities of the transcription convention, calculates as an output the additional (linguistic) structures of a Segmented-Transcription. In EXMARaLDA, such segmentation algorithms are implemented in the form of Finite State Machines. The advantage of this approach lies in its simplicity: because finite state processing is a plain and well-understood concept, the effort for constructing a new or modified segmentation algorithm (for a new or modified transcription convention) and for transferring a segmentation algorithm between different software applications[20] is minimal.
A detailed flow chart of the automatic segmentation process is given in the following figure:



```
<event start="T0" end="T1">Okay. </event>
<event start="T1" end="T2">Très bien, </event>
<event start="T2" end="T3">très bien. </event>
```

```
<segmentation name="SegmentChain_Event">
  <ts n="sc" s="T0" e="T3">
    <ts n="e" s="T0" e="T1">Okay. </ts>
    <ts n="e" s="T1" e="T2">Très bien, </ts>
    <ts n="e" s="T2" e="T3">très bien. </ts>
  </ts>
</segmentation>
```

Okay. Très bien, très bien.

```
<segmentation name="SegmentChain_Utterance_Word">
  <ts n="sc" s="T0" e="T3">
    <ts n="HIAT:u" s="T0" e="T1">
      <ts n="HIAT:w" s="T0" e="T1">Okay</ts>
      <nts n="HIAT:ip">.</nts>
    </ts>
    <ts n="HIAT:u" s="T1" e="T3">
      <ts n="HIAT:w" s="T1" e="T1.1">Très</ts>
      <nts n="HIAT:ip"> </nts>
      <ts n="HIAT:w" s="T1.1" e="T2">bien</ts>
      <nts n="HIAT:ip">,</nts>
      <nts n="HIAT:ip"> </nts>
      <ts n="HIAT:w" s="T2" e="T2.1">très</ts>
      <nts n="HIAT:ip"> </nts>
      <ts n="HIAT:w" s="T2.1" e="T3">bien</ts>
      <nts n="HIAT:ip">.</nts>
    </ts>
  </ts>
</segmentation>
```

Figure 6: Segmentation Process

[19] This section may actually seem irrelevant to the main topic of this paper because it is not about time based or hierarchy based data models, but about an application detail. However, as this particular application detail plays a key role for the manageability of the data structure, its principal mode of operation shall be outlined here.
[20] This is further facilitated by encoding the transition rules of the Finite State Machines in the form of an XML file that is then given as a parameter to a Finite State Machine Object implemented in JAVA.

Automatic Segmentation consists of three principal steps: Step 1 transforms a Basic-Transcription into a Segmented-Transcription with one segmentation in which consecutive events in tiers of type `T` are grouped into segment chains. Step 2 extracts the character data of these segment chains. In step 3, these character data are fed into a Finite State Machine whose output will then be a second segmentation of the same data.

This architecture for segmenting transcriptions has already proven its practicability: at present, the EXMARaLDA system contains Finite State Machines for segmentation according to four different transcription conventions[21] at least three of which are used in the every day work of linguists. [… to be continued…]

6. The TEI guidelines for the Transcription of Speech (P4)

In this section, I will briefly summarize what solutions the TEI offers for the encoding of transcriptions of spoken language and discuss some of the problems that these solutions may hold for the aim of this paper.

6.1. Representing temporal relations in TEI

As mentioned in the introductory section, the TEI data model is basically that of an ordered hierarchy, i.e. it builds on the assumption that the elements of a text – transcriptions are treated as texts of a special kind – can be brought into a meaningful sequence and be further structured into a single hierarchy in which smaller consecutive elements are grouped into larger elements. In this model, which is often referred to as an OHCO[22] model, parallel temporal relations have to be treated as an exception to the rule, and the TEI guidelines suggest a broad range of techniques for integrating these exceptions into the primary OHCO structure, for instance:

- A temporal overlap of two elements following one another in the document hierarchy can be encoded with the help of an attribute `trans` that characterizes the transition as an `overlap`:

```
<u who="A">I say something. </u>
<u who="B" trans="overlap">And I interrupt you. </u>
```

- For a more precise encoding of the extent of an overlap, the `synch` attribute can be used to mark the synchronicity of two elements in the hierarchy. The value of this attribute will then correspond to that of an `id` attribute of the complementary overlapping element:

```
<u who="A">I say <seg synch="u23">something. </seg></u>
<u who="B" id="u23">And I interrupt you. </u>
```

- Alternatively, the same relation can be expressed by the use of empty `<anchor>` elements which refer to one another via `synch` and `id` attributes:

```
<u who="A">I say<anchor id="a1" synch="b1"/>something. <anchor id="a2" synch="b2"/></u>
<u who="B"><anchor id="b1"/>And I interrupt you. </anchor id="b2"/></u>
```

[21] Beside the afore-mentioned HIAT (Rehbein et al. 2004), there are FSMs for the GAT (Selting et al. 1998) and DIDA (Klein/Schütte 2004) conventions, both used in conversation or discourse analysis in Germany, and for the CHAT system (MacWhinney 2000), used for child language acquisition studies.

[22] Ordered Hierarchy of Content Objects, cf. De Rose et al. (1990).

- A further method is the use of a `<timeline>` element grouping together a sequence of `<when>` elements. These `<when>` elements represent timepoints that can be referred to from elements in the document hierarchy via a `start` and an `end` attribute:

```
<timeline>
   <when id="t2"/>
   <when id="t3"/>
</timeline>
<u who="A">I say <seg start="t2" end="t3">something. </seg></u>
<u who="B" start="t2" end="t3">And I interrupt you. </u>
```

- Finally, for the special case of prosodic features, the TEI guidelines suggest the use of a `<shift>` element. This is an instance of a so called *milestone* element, i.e. an empty element which, instead of marking a stretch where a certain phenomenon occurs, rather marks the *point in time* at which the phenomenon starts. Thus, speaker DS's utterance in the above example could be encoded as follows:

```
<u>Okay.<shift feature="tempo" new="getting faster"/>Très bien, très bien.</u>
```

I will not discuss the benefits and drawbacks of each single of these diverse methods for encoding parallel temporal relations. Rather, I want to argue that their diversity can in itself be a problem for the processability and exchangeability of transcription data. Given that one and the same relation can be encoded in at least five different ways[23], constructing a software tool that actually makes use of this information becomes a very difficult task. Standard XML processing tools will not adequately support this task because they assume that the hierarchical document structure encoded in the nested elements is the paramount concern. Tools that operate on a time-based conception of data, on the other hand, will have difficulties extracting the temporal structures from this diversity of encoding techniques in a non-ambiguous way. The most important step in accommodating the TEI guidelines with time-based data models may therefore be a uniform approach to the encoding of temporal relations within a document hierarchy.

6.2. Transcription entities in the TEI

The main aim of the TEI guidelines is characterized in the following quote from Sperberg-McQueen/Burnard (2001):

> [The TEI Guidelines] provide means of representing those features of a text which need to be identified explicitly in order to facilitate processing of the text by computer programs. In particular, they specify a set of markers (or tags) which may be inserted in the electronic representation of the text, in order to mark the text structure and other textual features of interest.

Since transcriptions of spoken language are also treated as texts (though "texts of a special kind"), the guidelines also provide a set of tags specifically devised for the markup of the structure and features of interest of spoken language interactions. The most important of these are:

- The tag `<u>` for the an element which Johansson (1995: 87) defines as "[...] an utterance, i.e. a stretch of speech usually preceded and followed by silence or a change of speaker".
- A tag `<pause>` for encoding a pause

[23] And many more ways can be devised by combining these methods, e.g. by using a `synch` attribute inside a `<when>` element or by referring to a timeline from an `<anchor>` element.

- Tags **`<vocal>`**, **`<kinesic>`** and **`<event>`** for encoding non-lexicalized phenomena in a spoken interaction
- A **`<shift>`** tag for encoding prosodic phenomena

Furthermore, some of the tags defined in other sections of the TEI guidelines may also be relevant for the transcription of spoken language, in particular:

- The **`<w>`** tag for marking up individual words
- The **`<seg>`** tag for a subdivision of **`<u>`**-elements above the word level

The entirety of these elements make the TEI guidelines for Transcriptions of Speech a somewhat more concrete approach than most time-based data models: whereas especially the AG framework is very careful to introduce as few ontological specifications as possible in order not to jeopardize its broad applicability, the TEI guidelines are on a level of abstraction somewhere between a concrete transcription system like HIAT or CHAT and a general data processing framework like AG. That the TEI guidelines thus “prescribe” a general structure for the description of spoken language has been criticized, for instance by Sinclair (1995):

> I don’t have utterances as units in my descriptive system, and indeed many transcription systems don’t have rigorously defined utterances. [...] I will personally not accept TEI if it requires me to have an utterance under the definition that Lou Burnard was using, because that is far too rigorous for me and it doesn’t represent the world, as far as I’m concerned.

It is, however, indisputable that the main value of any approach to the encoding of transcriptions of spoken language will lie in its ability to find a convincing compromise between abstraction (ensuring its flexibility and broad applicability) and concreteness (ensuring its practicability and efficiency). As time-based models and the TEI guidelines seem to ascribe different weights to these issues, finding a bridge between the two approaches will also involve taking a decision on such a compromise.

7. A proposal for a 'TEI conformant' time-based data model

In this section, I will propose a data format that can function as a bridge between the time based EXMARaLDA data models[24] described in sections 3 and 5 and the hierarchy based TEI data model described in section 6. I have put the words 'TEI conformant' into inverted commas because, at one place, I will make a suggestion for a slight modification of a TEI element. The overall maxim, however, is to use as many of the existing TEI concepts as possible.
I have organized this section into a sequence of instructions that a transcriber who wants to create a TEI conformant transcription that is suited to be transformed to EXMARaLDA should be able to follow.

7.1. Basic Structure

- Describe the speakers in a **`<particDesc>`** element in the header

What is called the speakertable in an EXMARaLDA transcription unambiguously corresponds to the TEI element **`<particDesc>`** (section 23.2.2. of P4): both list the speakers participating in the interaction and provide them with IDs that can be referred to from elements in the transcription itself.

EXMARaLDA	TEI
<head>	<teiHeader>
	<profileDesc>
<speakertable>	<particDesc>
<speaker id="SPK0" abbreviation="DS"/>	<person id="DS"/>
<speaker id="SPK1" abbreviation="FB"/>	<person id="FB"/>
</speakertable>	</particDesc>
	</profileDesc>
</head>	</teiHeader>

- Use a **`<timeline>`** element to represent a single, fully ordered timeline

An equally clear-cut correspondence exists between the common timeline of an EXMARaLDA transcription and the TEI element **`<timeline>`** (section 14.5.2 of P4). The points on these timelines are represented as **`<tli>`** and **`<when>`** elements, respectively, and these, like the speakers, are given an **`id`** attribute that elements in the actual transcription can refer to.

EXMARaLDA	TEI
<common-timeline>	<timeline>
<tli id="T0"/>	<when id="T0">
<tli id="T1"/>	<when id="T1"/>
<tli id="T2"/>	<when id="T2"/>
<tli id="T3"/>	<when id="T3"/>
<tli id="T4"/>	<when id="T4">
<tli id="T5"/>	<when id="T5"/>
</common-timeline>	</timeline>

[24] I will neither restrict my considerations to the simpler Basic-Transcription data model nor will I consider the full complexity of the Segmented-Transcription data model. Rather, I will try to stick to the Basic-Transcription model as far as possible and use concepts from the Segmented-Transcription model only where I can think of no way to do without them. Section 8 will then make clear that this can nevertheless bring about a certain form of compatibility between EXMARaLDA and TEI data.

- ➢ Structure the main verbal flow of the interaction into **`<u>`** elements.
- ➢ Assign these **`<u>`** elements to the timeline via **`start`** and **`end`** attributes.
- ➢ Use an additional element **`<div type=‶segmental″>`** to group the segmental elements of an utterance beneath a **`<u>`** element.
- ➢ Represent additional temporal information within a **`<div type=″segmental″>`** element *exclusively* with the help of **`<anchor>`** elements.

The definition given in the TEI guidelines for the **`<u>`** element (section 11.2 of P4) is close to the EXMARaLDA definition of a segment chain. In particular, both are elements for a top level structuring of the lexical entities of an interaction and can thus form the superordinate node of a hierarchical representation of utterances, phrases, words, etc. It seems therefore reasonable to equate segment chains with **`<u>`** elements for the purposes of this paper.

The set of all **`<u>`** elements in a transcription can be brought into a meaningful sequence, for instance by ordering them according to their start points in the transcribed interaction. However, additional means to represent a partial or total overlap of different speakers' **`<u>`** elements have to be provided. This can be done by requiring an assignment of **`<u>`** elements to the timeline via **`start`** and **`end`** attributes. For the representation of additional temporal information – like a timepoint *within* a **`<u>`** element where another speaker's turn sets in – I suggest to uniformly use the empty **`<anchor>`** element (section 14.3 of P4) with a **`synch`** attribute referring to the timeline.

Beside the words (and, possibly, pauses and non-phonological elements, see below) that an utterance is made of, a transcription may contain information about prosodic features of these words or other additional annotations. In order to be able to clearly separate these different levels from one another, I suggest to group them under **`<div>`** elements (section 7.1.1 of P4) with appropriate values for the attribute **`type`**. For the **`<div>`** element grouping the actual words uttered, the value of this attribute could be **`segmental`**.

EXMARaLDA	TEI
<ts n="sc" s="T0" e="T3"> <ts n="e" s="T0" e="T1">Okay. </ts> <ts n="e" s="T1" e="T2">Très bien, </ts> <ts n="e" s="T2" e="T3">très bien. </ts> </ts>	<u who="DS" start="T0" end="T3"> <div type="segmental"> Okay. <anchor synch="T1"/> Très bien, <anchor synch="T2"/> très bien. </div> </u>

- ➢ Put **`<event>`**, **`<kinesic>`** and **`<vocal>`** elements within a **`<u><div type= ‶segmental″>`** element only if they are *alternative* to speech.
- ➢ Do not provide additional temporal information for such elements. Instead, use **`<anchor>`** elements (see above) before and after them if required.

Many transcription systems make a distinction between non-phonological (or semi-lexical or non-lexical) elements that are alternative to speech and non-phonological elements that accompany speech. The first are often regarded as directly belonging to a turn or an utterance in the same way that a word belongs to a turn or an utterance. Hence, it seems desirable to allow an integration of such entities – for which the TEI provides the elements **`<event>`**, **`<kinesic>`** and **`<vocal>`** – into the **`<u>`** element. Note that this integration is an option, not a requirement: if the transcriber prefers to treat all non-phonological elements independently of the words uttered, he can choose to put the elements in question outside (and on the same hierarchical level as) the **`<u>`** element[25]. If these elements are put inside a **`<u>`** element, they should not be given any additional temporal information (e.g. in the form of **`start`** or **`end`** attributes). This should help to avoid potential redundancies an to guarantee a uniform encoding of temporal relations.

EXMARaLDA	TEI
<ts n="sc" s="T2" e="T5"> <ts n="e" s="T2" e="T3">Alors ça </ts> <ts n="e" s="T3" e="T4">dépend ((cough)) </ts> <ts n="e" s="T4" e="T5">un petit peu. </ts> </ts>	<u who="FB" start="T2" end="T5"> <div type="segmental"> Alors ça <anchor synch="T3"/> dépend <vocal desc="cough"/> <anchor synch="T4"/> un petit peu. </div> </u>

- ➢ Put **`<event>`**, **`<kinesic>`** and **`<vocal>`** elements on the top level (along with **`<u>`** elements) if they *accompany* speech
- ➢ In that case, provide them with a **`start`** and an **`end`** attribute, and – if appropriate – with a **`who`** attribute.

Conversely, those non-phonological phenomena that are not an alternative, but an accompaniment to phonological entities *must* be encoded as independent elements outside **`<u>`** elements. Since they cannot, in this case, inherit the temporal features and speaker assignment of a parent element, this information has to be provided in the form of appropriate attributes. For events that cannot be assigned to a speaker (e.g. "telephone rings"), the attribute **`who`** can be left out.

EXMARaLDA	TEI
<event start="T2" end="T4">right hand raised</event>	<u who="DS" start="T0" end="T3"> [...] </u> <event who="DS" desc="right hand raised" start="T2" end="T4"/> <u who="FB" start="T2" end="T5"> [...] </u>

[25] For the example below, this alternative would mean splitting the **`<u>`** element in two **`<u>`** elements and put a **`<vocal>`** element between them:

```
<u who="FB" start="T2" end="T4">
    <div type="segmental">
        Alors ça
        <anchor synch="T3"/>
        dépend
    </div>
</u>
<vocal who="FB" desc="cough" start="T4" end="T4.1"/>
<u who="FB" start="T4.1" end="T5">
    <div type="segmental">
        un petit peu.
    </div>
</u>
```

However, this is not what is represented in the tier structure of the original example.

- Treat **`<pause>`** elements like **`<event>`**, **`<kinesic>`** and **`<vocal>`** elements.
- Qualify **`<pause>`** elements either by a **`dur`** or by a **`type`** attribute.

Even more than the transcription of non-phonological elements, the transcription of pauses is a matter of controversial debate in discourse analysis (cf., for instance, Kowal/O'Connell 2000). Here, too, it is desirable to be able to distinguish at least between pauses attributed to a speaker turn that are part of a **`<u>`** element and pauses between speaker turns that are not part of a **`<u>`** element. Concerning the **`who`**, **`start`** and **`end`** attributes of **`<pause>`** elements, the same rules should be applied as for **`<event>`**, **`<kinesic>`** and **`<vocal>`** elements. The actual description of the pause should be provided either by giving its duration in a **`dur`** attribute or by selecting one of the values **`short`**, **`medium`** or **`long`** for the **`type`** attribute.

- Use an additional element **`<div type=‶prosodic″>`** to group the non-segmental elements of an utterance beneath a **`<u>`** element.
- Use a **`<prosody>`** element – a modified version of the **`<shift>`** element – to represent prosodic features of an utterance beneath the **`<div type=‶prosodic″>`** element.
- Do not use this **`<prosody>`** element like a milestone, but provide it with a **`start`** *and* an **`end`** attribute.

In the EXMARaLDA typology, the descriptions of prosodic phenomena are a kind of annotation, i.e. they belong in a tier of type **`A`**, because they are not independent elements, but can only occur alongside segmental elements pertaining to the same speaker. For the same reasons, the descriptions of prosodic phenomena in a TEI document should be subordinated to a **`<u>`** element. This can be done by adding a second **`<div>`** division to the **`<u>`** element with the value **`prosody`** for the attribute **`type`**. Underneath this element, I suggest to retain the feature/value pairs provided in the TEI guidelines (section 11.2.6 of P4), but to encode them in empty **`<prosody>`** elements carrying a start and end attribute instead of treating them as milestone elements. In that way, they will integrate themselves more easily into the temporal logic of the rest of the document.

EXMARaLDA	TEI

<tier id="TIE2" speaker="SPK0" category="v" type="t">
    <event start="T0" end="T1">Okay. </event>
    <event start="T1" end="T2">Très bien, </event>
    <event start="T2" end="T3">très bien. </event>
</tier>
 	
<tier id="TIE1" speaker="SPK0" category="sup" type="a">
    <event start="T1" end="T3">faster</event>
</tier>
	<u who="DS" start="T0" end="T3">
	    <div type="segmental">
	        Okay.
	        <anchor synch="T1"/>
	        Très bien,
	        <anchor synch="T2"/>
	        très bien.
	    </div>
	    <div type="prosodic">
	        <prosody feature="tempo" desc="getting faster"
	        start="T1" end="T3"/>
	    </div>
	</u>

- Order top level elements
  1. by their start points (in increasing order)
  2. by their end points (in decreasing – "longer" elements first)
  3. by their type – <u> before others
  4. by the sequence of speakers

7.2. Additional Structure (to be elaborated)

Additional suggestions for additional annotations

- Allow <w> and <c> elements inside <u>s
- Allow <seg type="…"> inside <u>s
- allow additional <div type="…"> elements for additional annotations (other tiers of type t), subdivide them with the help of <seg> elements (???)

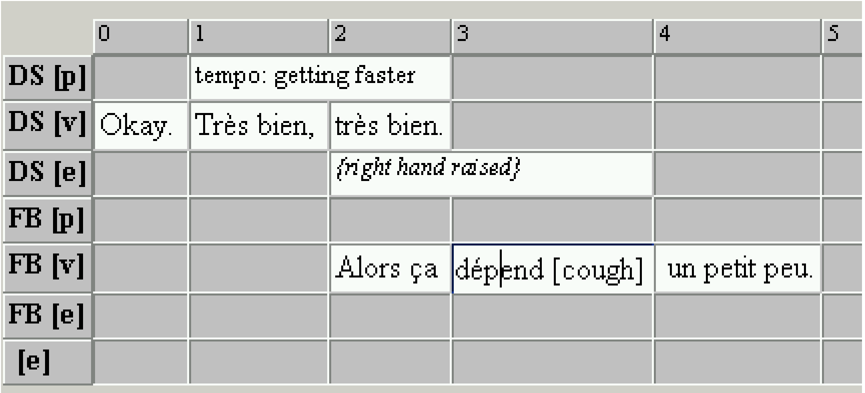
8. Application

As indicated in section 1, bringing together time based data models and the TEI should have a concrete practical value. In this section, I will demonstrate two scenarios of how a transcriber could profit from a compatibility between the two data models. Proof-of-concept versions of the corresponding conversion methods have been implemented and will be integrated into the next release of the EXMARaLDA Partitur-Editor.

8.1. Converting the TEI format to the EXMARaLDA Basic-Transcription format

If a transcriber has created a transcription according to the instructions given in section 7.1. (see Appendix D), he can use an import filter to transform this file into an EXMARaLDA Basic-Transcription. In that way, the following functionalities of this tool become available:

- The EXMARaLDA Partitur-Editor can be used to further edit the transcription in a musical score GUI, i.e. tiers and event descriptions can be added, deleted or changed. The result of these changes can be retransformed into a TEI format using the methods described in section 8.2



	0	1	2	3	4	5
DS [p]		tempo: getting faster				
DS [v]	Okay.	Très bien,	très bien.			
DS [e]			{right hand raised}			
FB [p]						
FB [v]			Alors ça	dépend [cough]	un petit peu.	
FB [e]						
[e]						

- The EXMARaLDA Partitur-Editor can be used to add further meta-data or speaker descriptions to the transcription.
- The output functionalities of the EXMARaLDA Partitur-Editor can be used to create different visualizations of the transcription in musical score or column notation (see section 2).
- The export functionalities of the EXMARaLDA Partitur-Editor can be used to further transform the data into another time based format like Praat, TASX, or ELAN.

The import filter is based on a SAX handler (written in Java) that parses the XML-coded TEI file and builds a Basic-Transcription Java Object from it. This involves the following transformations:

- The **`<particDesc>`** and **`<timeline>`** elements are mapped one-to-one on a speakertable and a timeline object of the EXMARaLDA transcription.
- Character Data between an opening **`<u>`** tag and an **`<anchor>`**, between two **`<anchor>`**s or between an **`<anchor>`** and a closing **`</u>`** tag are transformed into an event description. The corresponding event gets its start and end points from the temporal information provided by the attributes of **`<u>`** and **`<anchor>`** tags. The event is then added to a tier of type **`T`** (with the category 'v') of the associated speaker.
- The value of the **`desc`** (or the **`type`** or **`dur`**) attributes of **`<vocal>`**, **`<kinesic>`**, **`<event>`** and **`<pause>`** elements are also transformed into event descriptions. If these elements oc-

cur underneath a **`<u>`** element, the event descriptions are integrated into events in the appropriate tier of type **T**, otherwise they become independent events in a tier of type **D** (with the category ‘e’). In order to visually separate them from the lexical data, and to have an unambiguous implicit markup for the backwards transformation (see section 8.2.), these descriptions are enclosed in different types of brackets (square brackets for **`<vocal>`**, curly brackets for **`<event>`**, round brackets for **`<kinesic>`** and angle brackets for **`<pause>`**).

- The values of the **`feature`** and **`desc`** attributes of a **`<prosody>`** element are transformed into an event description by concatenating them with an intervening colon. The corresponding event takes over its start and end point from the **`<prosody>`** element. The event is then added to a tier of type **A** (with the category ‘p’) of the associated speaker.

8.2. Converting the EXMARaLDA Basic-Transcription format to the TEI format

The conversion between the two data formats in the other direction requires the transcriber to follow a set of simple conventions when creating or editing a transcription in the EXMARaLDA Partitur-Editor:

1. For each speaker, provide a tier of type **T** and category ‘v’ for the phonological (or lexical) elements.
2. If required, provide a tier of type **A** and category ‘p’ for the prosodic elements of each speaker.
3. If required, provide a tier of type **D** and category ‘e’ for the non-phonological (or non-lexical) elements of each speaker.
4. If required, provide an additional tier of of type **D** and category ‘e’ for the non-phonological (or non-lexical) elements that cannot be attributed to a particular speaker.
5. Put the descriptions of pauses and events and of vocalic and kinesic elements in a pair of brackets (square brackets for vocalic elements, curly brackets for events, round brackets for kinsesic elements and angle brackets for pauses).
6. The description of prosodic elements consists of two parts: the feature and its value. Separate the two with a colon, e.g.: *tempo: getting faster*

If a transcriber follows these conventions, he can use an export filter to transform the resulting EXMARaLDA Basic-Transcription file into a TEI file. In that way, he can profit from the functionality of tools that operate on a time-based data model and still produce ‘TEI-conformant’ data.

The conversion is mainly done with the help of built-in features of the EXMARaLDA system: The Basic-Transcription is first transformed into a Segmented-Transcription in which consecutive events in tiers of type **T** are grouped into segment chains. On the basis of these Segment-Chains, a List-Transcription is then calculated that constitutes a hierarchized version of a Segmented-Transcription (see appendix C). The List-Transcription is structurally already very similar to a TEI transcription such that, in a last step, it suffices to apply some XSL transformations to arrive at a TEI conformant document (see appendix D).

9. Conclusion and outlook

In this paper, I have suggested a method for bringing together the time based EXMARaLDA data model and an instance of the hierarchy based TEI data model. I have tried to demonstrate the practical value of such an undertaking by providing a proof-of-concept implementation of conversion filters between the data formats in question.

[… to be continued…]

Bibliography

Boersma, Paul / Weenik, David (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. (Updated copy of this manual at www.praat.org)

Brugman, Hennie / Russel, Albert (2004): Annotating Multi-media / Multi-model resources with ELAN. In: Lino, M. / Xavier, M. / Ferreira, F. / Costa, R. / Silva, R. (eds.): Proceedings of the Fourth International Conference on Language Resources and Evaluation. Paris: ELRA, 2065 – 2068.

De Rose, Steven / Durand, David / Mylonas, Elli / Renear, Allen (1990): What is Text, Really? In: Journal of Computing in Higher Education 1(2), 3-26.

Edwards, Jane / Lampert, Martin (eds.) (1992): Talking Data – Transcription and Coding in Discourse Research. Hillsdale: Erlbaum.

Edwards, Jane (1992): Principles and Contrasting Systems of Discourse Transcription. In: Edwards / Lampert (1992), 3-31.

Ehlich, Konrad (1992): HIAT - a Transcription System for Discourse Data. In: Edwards / Lampert (1992), 123-148.

Kipp, Michael (2001): Anvil - A Generic Annotation Tool for Multimodal Dialogue. In: Proceedings of of the 7th European Conference on Speech Communication and Technology, Aalborg, 1367 –1370.

Kowal, Sabine / O'Connell, Daniel (2000): Psycholinguistische Aspekte der Transkription: Zur Notation von Pausen in Gesprächstranskripten. In: Linguistische Berichte 183, 360 - 378.

Martinet, André (1960): Eléments de linguistique générale. Paris: Colin.

Milde, Jan-Torsten / Gut, Ulrike (2002): The TASX Environment: An XML-based Toolset for Time Aligned Speech Corpora. In: Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002), Gran Canaria.

Ochs, Elinor (1979): Transcription as theory. In: Ochs, Elinor / Schieffelin, Bambi (eds.): Developmental Pragmatics. New York, San Francisco, London: Academic Press.

Rehbein, Jochen / Schmidt, Thomas / Meyer, Bernd / Watzke, Franziska / Herkenrath, Annette (2004): Handbuch für das computergestützte Transkribieren nach HIAT. In: Working papers in Multilingaulism, Series B (Nr. 56). Universität Hamburg: Sonderforschungsbereich Mehrsprachigkeit.

Sasaki, Felix / Witt, Andreas (2004): Co-reference in Japanese Task-oriented Dialogues: A contribution to the Development of Language-specific and Language-general Annotation Schemes and Resources. In: Lino, M. / Xavier, M. / Ferreira, F. / Costa, R. / Silva, R. (eds.): Proceedings of the Fourth International Conference on Language Resources and Evaluation. Paris: ELRA, 655 – 658.

Appendix A: The example as an EXMARaLDA Basic-Transcription

```
<basic-transcription>
    <head>
        <speakertable>
            <speaker id="SPK0" abbreviation="DS"/>
            <speaker id="SPK1" abbreviation="FB"/>
        </speakertable>
    </head>
    <body>
        <common-timeline>
            <tli id="T0"/>
            <tli id="T1"/>
            <tli id="T2"/>
            <tli id="T3"/>
            <tli id="T4"/>
            <tli id="T5"/>
        </common-timeline>
        <tier id="TIE1" speaker="SPK0" category="sup" type="a">
            <event start="T1" end="T3">faster</event>
        </tier>
        <tier id="TIE2" speaker="SPK0" category="v" type="t">
            <event start="T0" end="T1">Okay. </event>
            <event start="T1" end="T2">Très bien, </event>
            <event start="T2" end="T3">très bien. </event>
        </tier>
        <tier id="TIE3" speaker="SPK0" category="en" type="a">
            <event start="T0" end="T1">Okay. </event>
            <event start="T1" end="T3">Very good, very good.</event>
        </tier>
        <tier id="TIE4" speaker="SPK0" category="nv" type="d">
            <event start="T2" end="T4">right hand raised</event>
        </tier>
        <tier id="TIE5" speaker="SPK1" category="v" type="t">
            <event start="T2" end="T3">Alors ça </event>
            <event start="T3" end="T4">dépend ((cough)) </event>
            <event start="T4" end="T5">un petit peu. </event>
        </tier>
        <tier id="TIE6" speaker="SPK1" category="en" type="a">
            <event start="T2" end="T5">That depends, then, a little bit</event>
        </tier>
        <tier id="TIE7" speaker="SPK1" category="pho" type="a">
            <event start="T4" end="T5">[ɛ̃tipø:]</event>
        </tier>
    </body>
</basic-transcription>
```

Appendix B: The example as an EXMARaLDA Segmented-Transcription (excerpt)

```
<segmented-transcription>
    […]
    <common-timeline>
        <tli id="T0"/>
        <tli id="T1"/>
        <tli id="T2"/>
        <tli id="T3"/>
        <tli id="T4"/>
        <tli id="T5"/>
    </common-timeline>
    […]
    <segmented-tier id="TIE_V_SPK0" speaker="SPK0" category="v" type="t">
        <timeline-fork start="T1" end="T2">
            <tli id="T1.1"/>
        </timeline-fork>
        <timeline-fork start="T2" end="T3">
            <tli id="T2.1"/>
        </timeline-fork>
        <segmentation name="SegmentChain_Utterance_Word">
            <ts n="sc" s="T0" e="T3">
                <ts n="HIAT:u" s="T0" e="T1">
                    <ts n="HIAT:w" s="T0" e="T1">Okay</ts>
                    <nts n="HIAT:ip">.</nts>
                </ts>
                <ts n="HIAT:u" s="T1" e="T3">
                    <ts n="HIAT:w" s="T1" e="T1.1">Très</ts>
                    <nts n="HIAT:ip"> </nts>
                    <ts n="HIAT:w" s="T1.1" e="T2">bien</ts>
                    <nts n="HIAT:ip">,</nts>
                    <nts n="HIAT:ip"> </nts>
                    <ts n="HIAT:w" s="T2" e="T2.1">très</ts>
                    <nts n="HIAT:ip"> </nts>
                    <ts n="HIAT:w" s="T2.1" e="T3">bien</ts>
                    <nts n="HIAT:ip">.</nts>
                </ts>
            </ts>
        </segmentation>
        <segmentation name="SegmentChain_Event">
            <ts n="sc" s="T0" e="T3">
                <ts n="e" s="T0" e="T1">Okay. </ts>
                <ts n="e" s="T1" e="T2">Très bien, </ts>
                <ts n="e" s="T2" e="T3">très bien. </ts>
            </ts>
        </segmentation>
    </segmented-tier>
    […]
</segmented-transcription>
```

Appendix C: The example as an EXMARaLDA List-Transcription

```xml
<list-transcription>
    <head>
        <speakertable>
            <speaker abbreviation="DS"/>
            <speaker abbreviation="FB"/>
        </speakertable>
    </head>
    <list-body>
        <common-timeline>
            <tli id="T0"/>
            <tli id="T1"/>
            <tli id="T2"/>
            <tli id="T3"/>
            <tli id="T4"/>
            <tli id="T5"/>
        </common-timeline>
        <speaker-contribution speaker="DS">
            <main>
                <ts n="sc" s="T0" e="T3">
                    <ts n="e" s="T0" e="T1">Okay. </ts>
                    <ts n="e" s="T1" e="T2">Très bien, </ts>
                    <ts n="e" s="T2" e="T3">très bien.</ts>
                </ts>
            </main>
            <dependent name="Event">
                <ats n="e" s="T2" e="T4">{right hand raised}</ats>
            </dependent>
            <annotation name="p">
                <ta s="T1" e="T3">tempo: getting faster</ta>
            </annotation>
        </speaker-contribution>
        <speaker-contribution speaker="FB">
            <main>
                <ts n="sc" s="T2" e="T5">
                    <ts n="e" s="T2" e="T3">Alors ça </ts>
                    <ts n="e" s="T3" e="T4">dépend [cough]</ts>
                    <ts n="e" s="T4" e="T5"> un petit peu.</ts>
                </ts>
            </main>
        </speaker-contribution>
    </list-body>
</list-transcription>
```

Appendix D: The example as TEI conformant document

```
<TEI.2>
    <teiHeader>
        <fileDesc/>
        <profileDesc>
            <particDesc>
                <person id="DS"/>
                <person id="FB"/>
            </particDesc>
        </profileDesc>
    </teiHeader>
    <text>
        <timeline>
            <when id="T0"/>
            <when id="T1"/>
            <when id="T2"/>
            <when id="T3"/>
            <when id="T4"/>
            <when id="T5"/>
        </timeline>
        <u who="DS" start="T0" end="T3">
            <div type="segmental">
                Okay.
                <anchor synch="T1"/>
                Très bien,
                <anchor synch="T2"/>
                très bien.
            </div>
            <div type="prosody">
                <prosody feature="tempo" desc="getting faster" start="T1" end="T3"/>
            </div>
        </u>
        <event who="DS" desc="right hand raised" start="T2" end="T4"/>
        <u who="FB" start="T2" end="T5">
            <div type="segmental">
                Alors ça
                <anchor synch="T3"/>
                dépend
                <vocal desc="cough"/>
                <anchor synch="T4"/>
                un petit peu.
            </div>
        </u>
    </text>
</TEI.2>
```

Appendix E: Resources

Apart from the file formats exemplified in the previous appendices, this paper mentions some additional resources that play a role in the conversion between TEI and EXMARaLDA files:

- SAX Handler for conversion of a TEI transcription to an EXMARaLDA Basic-Transcription (written in JAVA)
- Document Type Definitions for EXMARaLDA Basic-, Segmented- and List-Transcriptions
- Algorithms for converting between EXMARaLDA Basic-, Segmented- and List-Transcriptions
- XSL Stylesheet for conversion of an EXMARaLDA List-Transcription to a TEI transcription

All of these have been integrated into version 1.3. (release date 01 Sep 2004) of the EXMARaLDA Partitur-Editor which can be freely downloaded from

http://www.rrz.uni-hamburg.de/exmaralda

Please contact the author (thomas.schmidt@uni-hamburg.de) for a copy of the source code or additional documentation not available on the website.