Visualising Linguistic Annotation as Interlinear Text

Thomas Schmidt
Sonderforschungsbereich "Mehrsprachigkeit"
University of Hamburg, Germany
thomas.Schmidt@uni-hamburg.de
http://www.rrz.uni-hamburg.de/exmaralda

Introduction

Interlinear Text (IT) is a widely used method of data visualisation in linguistics. In spite of this fact, and although there are quite a number of tools for inputting and outputting such data, IT has rarely been described from a formal point of view. This paper tries to do this by

a) showing where (in linguistics) IT is used,
b) attempting a characterisation of what IT is, and
c) outlining what may be necessary in order to work with IT

Section 1 gives examples of uses of IT in discourse transcription and other linguistic domains. In Section 2, IT is characterised as a method for visualisation of relations between textual items, combining properties of a table with properties of linear text. Section 3 then discusses several aspects of the requirements for working with IT. Finally, section 4 briefly demonstrates how IT is used in the EXMARaLDA system.

1. Where is Interlinear Text used?

In Bird et al. (2002), IT is characterised as

> "a kind of text in which each word is annotated with some combination of phonological, morphological and syntactic information (displayed under the word) and each sentence is annotated with a free translation".

While this is an accurate description of the way IT is used in field linguistics, it does not capture other uses of IT, especially in the transcription of spoken language, nor does it make clear that IT can be used as a visualisation method in a much more general way, independently of the specific units word, sentence, free translation etc. mentioned here. Before attempting a more comprehensive description of what IT is, I therefore want to show in the following two sections for what other purposes IT is used.

1.1. Visualisation of temporal relations in the transcription of spoken language

Conventional written text is organised in a one-dimensional space, with characters running in one line (from left to right, right to left or top to bottom, depending on the writing system). This is enough to express sequential relations, i.e. one word following another or one sentence preceding the next. However, in spoken language transcription (which is often characterised as some kind of mapping from spoken language to a written text), it may be necessary to express that the temporal relation between two given items is parallel, not sequential. Most prominently, this is the case when two speakers' contributions overlap. Every transcription system has some way for visualising this parallel structure, and in the vast majority of cases,

IT plays some role in it. A transcription system that uses IT as the basic organisational component of its visualisation is HIAT[1] (Ehlich/Rehbein 1976, Ehlich 1992). In reference to the notation of musical instruments in an orchestral score, the authors speak of interlinear text as "musical score notation" (German: *Partiturnotation*). The following is an example from Ehlich (1992: 130):

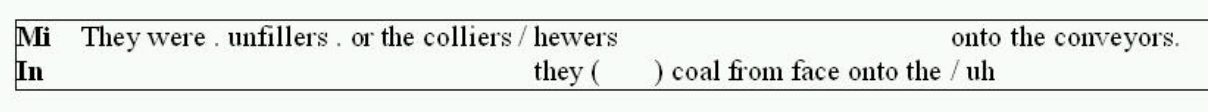

Figure 1

In this example, the fact that certain portions of speaker Mi's speech ('hewers' and 'onto') overlap with certain portions of speaker In's speech ('they' and 'uh' respectively) is represented by the alignment of these items on the same horizontal position. HIAT uses this extension of the one-dimensionality of written text to a second dimension not only to represent speaker overlap, but also to represent the temporal relation between the transcription of verbal data and data from other modalities like gesture, facial expression etc. (Ehlich/Rehbein 1979a), as in the following (made-up) example[2]:

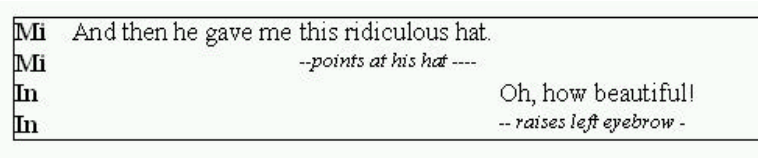

Figure 2

HIAT uses musical score notation throughout the whole transcript, i.e. even in passages where there is neither speaker overlap nor any other kind of parallel structure: The following example from Ehlich (1992: 130) demonstrates this:

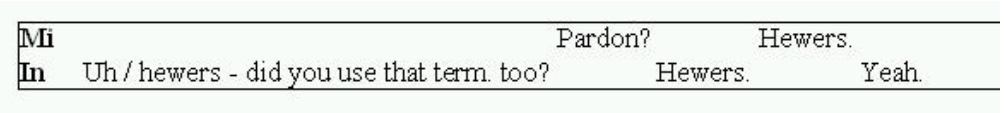

Figure 3

The authors argue that this consistent graphical organisation of the transcript improves its readability. Furthermore, it does not – as most vertically organized transcription systems do[3] – force the transcriber to segment the discourse into utterances or other (non-temporal) units.
There are few other transcription systems that use IT in the same rigid manner (the only ones I know of are DIDA (Klein/Schütte 2000) and the system described in Henne/Rehbock (2001)[4]). Most other transcription systems are organized in a way that Edwards (1992) calls

[1] As Edwards (1992: 12) puts it, the "disadvantage [of partiture format, T.S.] is its special requirements for alignment of items when the transcript is corrected or modified, which however can be mitigated by specialized computer programs." HIAT transcriptions are therefore usually made with such specialised software – syncWriter (Walter 1990, Dybkjaer et al. 2001) for the Macintosh and HIAT-DOS (Ehlich 1992) for DOS and Windows systems.
[2] The alignment of descriptions of non-verbal and verbal actions is also extensively used in STAVIS (Balthasar 2001).
[3] I use the term 'vertical' in the sense of Edwards (1992) – see below.
[4] Edwards (1992) also mentions Ervin-Tripp (1979) and Tannen (1984) as systems using "partiture" notation.

"vertical", i.e. they depart from the assumption that discourse structure can be *primarily* characterised as a sequence of speakers' contributions, and this sequence of contributions can be graphically represented in the conventional line-for-line notation. Only if an overlap occurs does this notation require a modification. In most cases, such systems express parallel temporal structure by some kind of bracketing, as in the following example from Du Bois et al. (1992: 77):

R: When he was real little,
[he] almost died of pneumonia.
L: [Oh].
R: when he was &

Figure 4

The temporal flow in this graphical representation is basically like the flow in a conventional written text: within a line, time flows from left to right, and lines that are further down on the page describe events that happened later in the discourse. The brackets indicate an exception to that principle – they state that in this position, the temporal flow is halted, and there are elements further down (or further up) on the page describing events that happened at the same time. Many systems contain additional instructions for avoiding potential ambiguities in cases where several speakers' contributions overlap at a time (like indexing brackets, using more than one bracket etc., cf. for instance Du Bois et al. 1992) so that this kind of system would probably be sufficient to express all possible kinds of parallel relations in a discourse. However, a large number of these systems[5] suggest that, mainly for improvement of readability, overlapping stretches of speech be also aligned horizontally, as in the following example (Du Bois et al. 1992: 52):

K: Cytomegalos [virus],
G: [Don't] forget,

Figure 5

Hence, although IT is not the primary principle of graphical organisation in these systems, it is (optionally) used whenever parallel structures occur.

1.2. Visualisation of equivalence relations in the analysis of spoken or written language

In the above examples from discourse transcriptions, IT is used to express temporal relations between descriptions of (verbal or non-verbal) events. The second wide-spread way of using IT in linguistics is not concerned with temporal relations, but rather with what I would preliminarily call *equivalence relations*. It is this use that underlies the characterisation given in the quote from Bird et al. (2002) above. In the following example (Jacobson et al. 2000), IT is used to align equivalent units of analysis:

[5] GAT (Selting et al. 1998) and CHAT (MacWhinney 2000), among others.

Figure 6

The graphical organisation expresses that the word 'nakpu' in the first line is equivalent to the English word 'two', that the sentence 'nakpu nonotso…' is equivalent to the English sentence 'They say…' and so forth. The term *equivalence*, in this context, is to be understood in a very broad sense. It is only intended to indicate that the relation between two items A and B at the same horizontal position is better described by 'A *is* B' rather than 'A and B *happen at the same time*', as in the examples in 1.1. This manner of using IT can be found in quite a few subdomains of linguistics. In field linguistics, its main purpose is probably to make accessible samples and analyses of little known languages to a research community that does not necessarily speak these languages, which is why such data often contain word glosses and translations. It may be used for the same purpose in discourse transcription, as in the following example (Rehbein et al. 1992: 105)[6]:

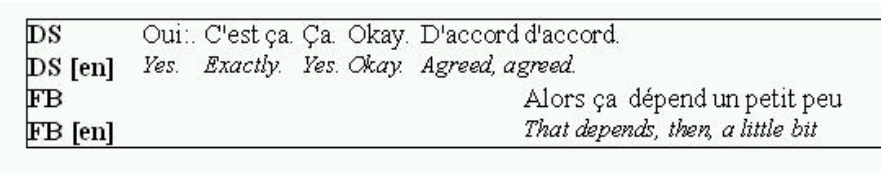

Figure 7

Ehlich (1992: 136) demonstrates another kind of use in HIAT discourse transcriptions, namely for the visualisation of intonation contours[7]:

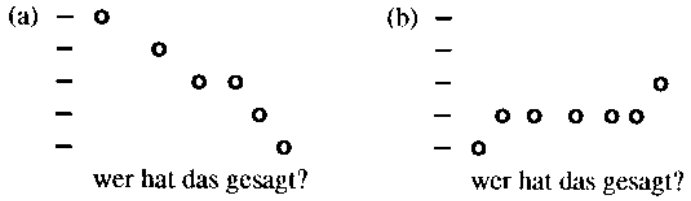

Figure 8

Again, this principle of graphical organisation cannot only be found in transcription systems with partiture notation, but also in transcription systems that use vertical notation. Consider the following examples from CHAT (MacWhinney 2000):

[6] I am not, by the way, suggesting that French is a little known language.

[7] The notion of equivalence seems a bit problematic in this case. It would probably be more appropriate to speak of an element-feature relation for such uses of IT, i.e. in the given example, the graphical representation expresses that a certain syllable element has a certain intonational feature. However, I think that for the purposes of this paper such element-feature relations can be subsumed under the notion of equivalence relations.

```
*MOT: yo no tengo nada.
%eng: I don't have anything.

*SAR: I got a boo+boo.
%pho: /ai gat V bubu/

*MOT: well go get it !
%mor: ADV|well V|go&PRES V|get&PRES PRO|it!
```

Figure 9

Although these examples do not make extensive use of interlinear alignment (the morphological information in the last example, for instance, is not aligned word for word, but rather for the entire utterance), they are examples of interlinear text in so far as their graphical organisation expresses equivalence relations *between lines.*

It becomes clear from these examples that there is no reasonable way to restrict the uses of IT to any specific linguistic levels like morphology, phonology and so on. As Sprouse (2000) puts it,

> "At a minimum, the model will accommodate all those levels found in the existing IT applications: [follows a list of levels]. *Obviously, this list cannot be comprehensive.*"

a simple enumeration of a (however large) number of levels cannot possibly be a sufficient description of what IT is.

2. What is Interlinear Text?

Following the examples given above, I would suggest the following characterisation of IT:

> "Interlinear Text is a form of graphical organisation of text where horizontal alignment of textual items on a number of consecutive lines is used to express temporal or equivalence relations."

In this view, IT is not so much a data model in itself, but rather a *visualisation method* for data models. In the same way as a table or the drawing of a mathematical graph, it is not very well suited to describe the abstract properties of a model, but rather to provide a concrete comprehensive and readable visualisation of such properties.
In order to better understand IT, it may therefore be useful to compare it to other forms of graphical organisations of textual items. Leaving more unusual ways like calligrams or other forms of word art aside, there are basically two ways of graphically organising textual items: one-dimensionally as a (linear) text or two-dimensionally as a table.
As outlined above, linear text predominantly encodes sequentiality. The 'meaning' of any textual item in a linear text is established via its position in the sequence of other textual items. Changing the sequence of items by exchanging the position of two of them or leaving one out fundamentally alters the meaning of the overall text. Other aspects of the graphical organisation like, for instance, the relation of the vertical positions of two items, on the other hand, are irrelevant to the meaning of the text. The following two examples therefore 'mean' the same thing:

I saw the man in the park with the telescope. He was wearing a blue hat.

I saw the man in the park with the telescope. He was wearing a blue hat.

Figure 10

A relation that can be derived from the sequential relation is that of containment. Besides telling the reader that certain textual items precede or follow certain other textual items (the word 'park' precedes the word 'with'), a text also tells him that certain textual items are contained in certain other textual items (the word 'hat' is contained in the sentence 'He was wearing a blue hat.').
A two-dimensional table, on the other hand, primarily neither visualises sequential relations nor containment. The 'meaning' of a textual item in a table is established with respect to its horizontal and vertical position, i.e. via the entries in the corresponding column and/or row header. Exchanging the position of two entire rows or columns therefore does not fundamentally alter the meaning of the overall table - the following two examples again 'mean' the same thing:

	John	Paul
Mother	Cynthia	Mary
Father	James	Frank

	Paul	John
Father	Frank	James
Mother	Mary	Cynthia

Figure 11

Given the versatility of tables, it is difficult to find a general term describing the semantics of the relation expressed in this form of graphical organisation. Here again, the notion of 'equivalence' may be a preliminary attempt to cover all potential uses (the above example, for instance, expresses that John's mother *is* Cynthia).
These observations about text and tables may seem somewhat trivial. However, I think they can serve to explain why IT is such a wide-spread method for visualising linguistic models: IT conveniently combines the method of text for visualising sequential and containment relations with the method of tables for visualising equivalence relations, and this combination of sequential, inclusion and equivalence relations (with the added possibility to express parallel relations, see above) is exactly what most linguistic models are concerned with.
In IT, the meaning of a textual item is established with respect to its position in a sequence of other textual items as well as with respect to its vertical and/or horizontal position, as in the following example:

John	Their	mother	used	to	smoke	cigars.
[POS]	DET	N	V	P	V	N

Figure 12

The textual item 'Their mother used to smoke cigars.' can be read like a normal one-dimensional text – the graphical organisation makes clear that the word 'mother' follows the word 'their', that both these words are contained in the entire utterance and so forth. At the same time, aspects of the meaning of these and other textual items can be established by reading the example like a table: The fact that the utterance is made by John, that the word

'mother' is classified as a noun, etc. are visualised, as in a table, by the respective horizontal and vertical arrangement of the textual items describing the corresponding entities.

3. Requirements for working with IT

3.1. What linguistic models can be visualised as IT?

In the preceding section, it was argued that IT is a visualisation method for (predominantly, but not necessarily linguistic) data models. In the same line of argument, I would suggest that the first question in dealing with IT is not "What is a (or the) data model for IT?", but rather "What data models can be visualised as IT?". Bird/Liberman (2001) have shown that a very large number of linguistic data models can, on a logical level, be expressed in the framework of annotation graphs (AG). i.e. as a set of directed, acyclic graphs whose nodes can be (partially) ordered according to one or several timeline(s) and whose arcs carry the non-temporal information of the data model. Building on that observation, the above question can be reformulated as "Which subclass of AGs can be visualised as IT?"
Maeda/Bird (2000) (who, in a way, approach this question from the opposite angle, namely by answering the question 'How can a model which is visualised as IT be formulated as an AG?') describe "structural limitations in Interlinear Text" and, departing from these limitations, formulate four conditions that an AG has to fulfil in order to "have an interpretation as interlinear text". I will briefly paraphrase these limitations and conditions here:

- The basic underlying assumption is that arcs in AGs have types ("each arc is ordinarily given a distinguished *type* attribute"). Condition 1 states that these types must fall into groups ("Every type belongs to exactly one group"). This condition reflects the fact that IT is organised in several lines – each of these lines usually contains descriptions of a certain linguistic type (like surface word form, lemma, part of speech etc.), and these descriptions (or lines) can be grouped into units that have the same alignment behaviour.
- Condition 2 states that arcs belonging to the same group must share the same structure, i.e. that arcs of the same group that share an end node must also have identical start nodes and vice versa. Condition 3 states that if a subgraph of a certain group A is contained in another subgraph of another group B, then group B must also contain group A[8]. These conditions reflect what the authors call "structural identity" (elements belonging to the same group are "always aligned") and "containment structure" respectively.
- Condition 4 finally requires that there be one group containing all other groups, i.e. that there be something like a top level category which dominates all other categories. In the given examples, this top level group usually corresponds to the sentence level which "contains" all other levels like the word level, the morphological level and the phonemic level.
- A restriction that is not explicitly formulated as a condition, but mentioned as a structural limitation of interlinear text is that "overlaps of arcs representing the same linguistic level of information […] should not occur".

The authors thus demonstrate that "the annotation graph model with conditions for interlinear texts can represent properties of interlinear text", or, in the reversed view: AGs that fulfil these conditions can be visualised as IT. I first want to show that these restrictions are too strong to cover all uses of IT, especially that the restrictions concerning groups are not

[8] There is no formal definition of containment of groups.

necessary and do not hold for transcriptions of spoken language. Consider the following excerpt of an IT transcription (cf. figure 7, the grey lines indicate positions of alignment):

	0	1	2
DS	D'accord	d'accord.	
DS [en]	Agreed, agreed.		
FB		Alors ça	dépend un petit peu
FB [en]		That depends, then, a little bit	

Figure 13

An intuitive representation as an AG would look like this:

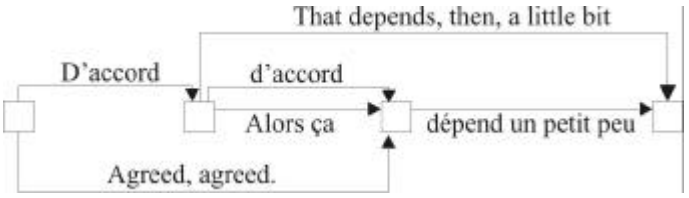

Figure 14

Nodes: n_0, n_1, n_2, n_3
Time function: $\tau(n_0) < \tau(n_1) < \tau(n_2) < \tau(n_3)$

Arcs:

start	end	Label	type	speaker	speaker/type	(group)
n_0	n_1	D'accord	verbal	DS	DS-v	A
n_1	n_2	d'accord	verbal	DS	DS-v	A
n_0	n_2	Agreed, agreed.	translation	DS	DS-tr	B
n_1	n_2	Alors ça	verbal	FB	FB-v	C
n_2	n_3	dépend un petit peu	verbal	FB	FB-v	C
n_1	n_3	That depends, then, a little bit	translation	FB	FB-tr	D

It seems natural to classify arcs not only according to types, but also according to speakers. However, as the above table indicates, these two features can be combined into a single speaker/type-feature and thus yield a basis for the required grouping. In the given example, the number of groups would probably be equal to the number of speaker/type combinations – one group *A* for verbal events of speaker DS, one group *B* for translations of speaker DS and corresponding groups for speaker FB. The grouping, in this case, therefore provides no additional information. Condition 4, furthermore, is not met by this AG – none of the four groups contains all other groups (group *B* contains group *A*, and group *D* contains group *C*, but there is no group that contains both *B* and *D*).
As, however, the AG clearly does have an interpretation as an IT, I would suggest that the following is a more comprehensive answer to the question "Which AGs can be visualised as IT?"[9]:

[9] I am aware that these conditions largely overlap with the ones given in Maeda/Bird (2000).

An annotation graph $G = <N, A, \tau>$ can be visualised as interlinear text if:

1. there is a partition $A = A_1 \cup A_2 \cup \ldots \cup A_n$ ($A_i \subseteq A$, $A_i \cap A_j = \emptyset$ for $i \neq j$) of arcs (according to their types[10])
2. for any A_i in this partition: two arcs $a_n, a_m \in A_i$ do not overlap
3. $\tau(n)$ is defined for all $n \in N$
4. $\tau(n_1) = \tau(n_2) \rightarrow n_1 = n_2$ for all $n_1, n_2 \in N$

Conditions 1 and 2 ensure that all arcs can be distributed onto a finite number of layers, and that in any one of these layers, arcs do not overlap. Conditions 3 and 4 require that all nodes be brought into an unequivocal temporal order with no two nodes representing the same point in time. Constructing an IT from a thus restricted AG is straightforward: the IT will have as many lines as there are elements in the partition, and labels of arcs sharing the same start node will be aligned.

I think that these conditions are the only *necessary* prerequisites for visualising an AG as IT. However, there is a further aspect that is not strictly required for constructing an IT from an AG, but that is nevertheless crucial in the use of IT. It is, in my opinion, this aspect that motivates the definition of groups in Maeda/Bird (2000): As outlined above, IT works well as a visualisation method for linguistic data because it makes use of certain properties of linear text like its ability to visualize sequentiality and containment. I would argue that, without these properties, IT would not be considered an economic and readable representation of linguistic annotation. What is exploited in IT is that many labels used to describe temporally anchored linguistic properties of a signal have themselves an *inherent temporal structure*. Because of this property, such labels can be meaningfully joined or split. A reader of an IT performs such split and join operations mentally when he looks at IT. I will call these properties *segmentability* and *combinability* and define them with the help of the AG framework:

Let $a = <n_1, n_2, l>$ be an arc with start node n_1 and end node n_2 ($\tau(n_1) < \tau(n_2)$) that appropriately describes some property of the underlying signal. Let l be the label of this arc and $s_0 s_1 \ldots s_n$ be the sequence of symbols that this label is made of (in other words: l is a string, and s_i are the characters of this string)

The arc a is **segmentable** if there is a node n_3 with $\tau(n_1) < \tau(n_3) < \tau(n_2)$ and a j with $0 < j < n$ such that the arcs $a_1 = <n_1, n_3, s_0 \ldots s_j>$ and $a_2 = <n_3, n_2, s_{j+1} \ldots s_n>$ also appropriately describe properties of the underlying signal.

Two arcs are **combinable** if the reverse is true, i.e. if for two arcs $a_1 = <n_1, n_3, s_0 \ldots s_j>$ and $a_2 = <n_3, n_2, s_{j+1} \ldots s_n>$ that appropriately describe properties of the underlying signal, the combined arc $a = <n_1, n_2, s_0 \ldots s_n>$ also appropriately describes a property of the underlying signal.

It is important to note that not all possible arcs are segmentable or combinable. For instance, in Figure 2, there is an event described by an arc carrying the label "points at his hat". Splitting this arc in two does not yield any appropriate description of parts of this event – e.g., the event is not made up of two consecutive events that can be described with "points" and "at his hat" in the same way that the event described by "Then he gave me this ridiculous hat." is made up of two consecutive events that can be described by "Then he gave me" and "this ridiculous hat". Although AGs that do not contain segmentable or combinable arcs can be visualised as IT (if they fulfil the requirements above), this IT will probably not be considered a "good" visualisation:

[10] The partition will naturally be in some way related to a semantic distinction between different arcs. However, it is by no means a necessary condition that each partition correspond to exactly one type of arc. For instance, arcs describing non-verbal behaviour of one and the same speaker may well be distributed over more than one layer (i.e. more than one partition), and one layer may contain arcs of different types as long as these do not overlap.

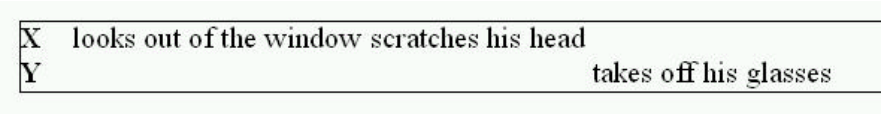

Figure 15

Furthermore, if there are no segmentable arcs, it is not possible to meaningfully align descriptions of temporally overlapping events in a way that exactly specifies the beginning and end of the overlap (grey lines indicate positions of alignment):

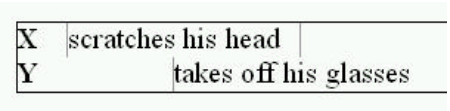

Figure 16

It is not only descriptions of non-verbal events that lack the property of segmentability and combinability. The same holds, for instance, for the part of speech tags in figure 12 (neither segmentable nor combinable) and the utterance translations in figures 6 and 7 (combinable, but not segmentable). However, in all examples of IT, wherever there is such a non-segmentable description, there also seems to be an associated segmentable description with which it is aligned.
Consequently, in order to have a "good" visualisation as an IT, an AG must fulfil a fifth condition, namely

> 5. in at least one A_i of the partition of A, arcs must be segmentable, and for each non-segmentable arc $a = <n_1, n_2, l>$, there is at least one segmentable arc b with the same start node n_1 and at least one segmentable arc c with the same end node n_2 (b and c may be identical).

This condition need not be met in all cases, but the more often it is not met, the less the IT will fulfil its purpose, i.e. the less it will be a readable visualisation of the underlying data model. Typically, the segementable set(s) of arcs referred to here will not correspond to a specific linguistic level, but rather to *several* linguistic levels. Every character of a segmentable description is a potential point of alignment in the IT (or a potential node in the underlying AG). In most cases, these characters will correspond to something like phonemes (at least in spoken language transcription), but not every phoneme will have its 'own' arc (consider, for example, figure 14).

3.2. A formal description of IT

If IT is a visualisation method for data models rather than a data model in its own right, a formal description of IT should be concerned with graphical properties rather than with logical structure. Here again, a look at the use of tables may illustrate my point. As outlined above, tables can be used to visualise a very wide range of data models; they can express a large number of logical relationships. Many document description languages – like HTML, LATEX, RTF, PDF, XSL:FO etc. - therefore provide formalised descriptions of tables. None of these languages, however, attempts to grasp any of the logical structure of a table[11]. They all restrict themselves to aspects of its graphical appearance, like the definition of rows, columns, cells, labels, cell spans, widths and heights, fonts, colours and so forth. I think a useful formal description of IT should do the same, i.e. it should concentrate on the graphical

[11] See Wohlberg (1999) for a work that tries to do this, i.e. describe a generalised logical structure of tables.

appearance of IT and leave the logical structure that can be implied from this appearance aside. It would then be up to the underlying data models to provide the logical description.

3.2.1. Basic level

At a most basic level, a formal description of IT will reflect the definition given above, i.e. the fact that any IT consists of several lines and that items on these lines may be aligned. An IT therefore consists of a number of alignment points and a number of lines, and these lines in turn consist of a number of items that are aligned with respect to the alignment points. The following figure (based on figure 12) illustrates this:

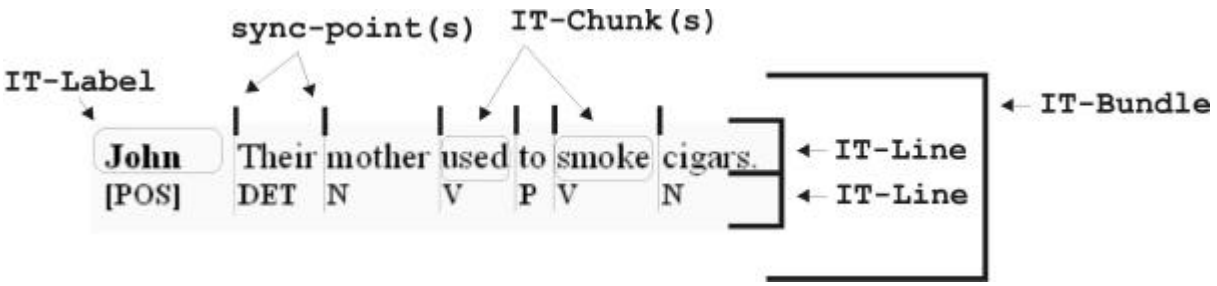

Figure 17

Formulated as an XML document type definition (DTD), this structure would look like this.

Elements	Attributes
`<!ELEMENT it-bundle (sync-point*, it-line*)>`	
`<!ELEMENT sync-point EMPTY>`	`<!ATTLIST sync-point id ID #REQUIRED>`
`<!ELEMENT it-line (it-label?, it-chunk*)>`	
`<!ELEMENT it-label #PCDATA>`	
`<!ELEMENT it-chunk #PCDATA>`	`<!ATTLIST it-chunk sync-start IDREF #REQUIRED>`

In the syntax of this DTD, the example can be expressed with the following XML document.

```
<it-bundle>
   <sync-point id="s0"/>
   <sync-point id="s1"/>
   <sync-point id="s2"/>
   […]
   <it-line>
      <label>John</label>
      <it-chunk sync-start="s0">Their</it-chunk>
      <it-chunk sync-start="s1">mother</it-chunk>
      <it-chunk sync-start="s2">used</it-chunk>
      […]
   </it-line>
   <it-line>
      <label>[pos]</label>
      <it-chunk sync-start="s0">DET</it-chunk>
      <it-chunk sync-start="s1">N</it-chunk>
      <it-chunk sync-start="s2">V</it-chunk>
      […]
   </it-line>
</it-bundle>
```

3.2.2. Formatting properties

The above is a sufficient description of the 'essence' of IT. An application provided with these data would in principle be able to construct the above graphical representation. It could choose an appropriate font (i.e. one that has glyphs for all the PCDATA) and, from the metrics of that font, calculate where to position the textual items on a screen or a print-out.

However, in many actual uses of IT, the chosen fonts, font sizes and styles, etc. may be important for the quality of the visualisation. The following examples illustrate that:

- Frequently, a description on a certain layer systematically requires fewer or more symbols than an associated description on another layer. For instance, POS tags are usually made up of fewer symbols than the words they refer to. In the following example, it is English utterances and their German translations that differ in the amount of symbols required to describe them – the translation always seems to be a bit "longer" than the utterance it refers to. If the same font is used throughout the entire IT, this results in gaps in the "main" line making the IT less readable (the graphical appearance may give the reader the impression that there are pauses between Max's utterances):

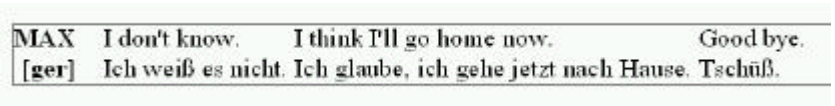

Figure 18

Choosing a smaller font for the translation mitigates this problem. Putting it in italic style can further serve to visually distinguish it from the transcription of actual utterances:

Figure 19

- In the description of non-verbal phenomena, there is no correlation of the temporal extension of an event and the typographic extension of the string that describes it (cf. also figure 2 – the dashes are not part of the actual description, they are just there to make the typographical extension of the description equal to the temporal extension of the described). Using additional formatting, like vertical lines or shading can make the extension of such phenomena clearer:

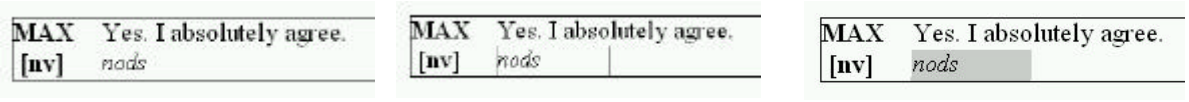

Figure 20

- Finally, additional formatting of (sequences) of characters can simply be used to express additional meaning. In HIAT, underlining sequences of characters is used to express intonational stress (example from Rehbein et al. 1992: 35).

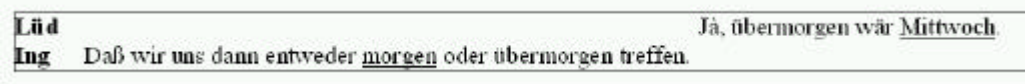

Figure 21

Other transcription systems use bold or italic print for the same purpose. Further examples are the use of increased letter spacing for slowly spoken passages or the use of different font colours for representation of different languages in multilingual talk.

Formatting properties are therefore a useful extension to the IT-Chunks defined above: One IT-Chunk can contain several differently formatted character sequences. Such differently formatted character sequences are often called *runs*. In order to integrate the formatting properties into the formal description of IT, the above DTD could be altered as follows:

Elements	Attributes
<!ELEMENT it-label (run+)>	
<!ELEMENT it-chunk (run+)>	<!ATTLIST it-chunk sync-start IDREF #REQUIRED sync-end IDREF #IMPLIED>[12]
<!ELEMENT run (format*, content)>	
<!ELEMENT format (#PCDATA)>	<!ATTLIST format property-name CDATA # REQUIRED>
<!ELEMENT content (#PCDATA)>	

The definition of the <format> element is done in a deliberately open manner, as an attribute/value pair, because it is not easily foreseeable which formatting properties one wants to specify. In practice, the attribute names and their possible values would have to be restricted by a closed vocabulary that an application can interpret[13].

3.2.3. Further extensions of the description

There are surely many more useful ways to further extend this formal description of IT. I would like to only briefly hint at three such extensions:

- Integration and alignment of image data

It may be useful to integrate image data into the IT, for instance in order to illustrate transcription of gestures, and align these data with corresponding textual data:

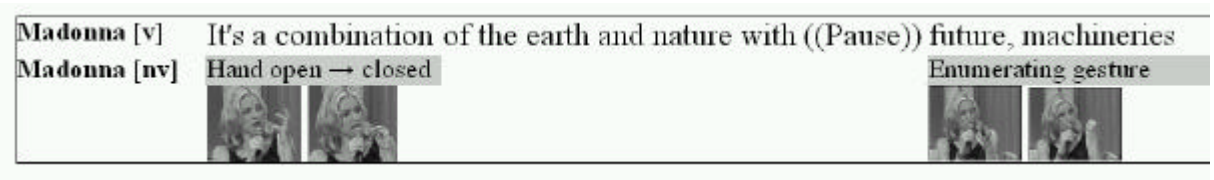

Figure 22

- Breakable IT-Chunks

The characters of an IT-Chunk will usually be aligned on an uninterrupted line. This improves the readability for most types of descriptions and is probably compulsory for segmentable and combinable descriptions. However, for some purposes, the opposite may be the case, i.e. it may improve the readability if the horizontal space required by a certain textual item is reduced by breaking the item up into several lines. One example is a layer in spoken language transcriptions where a transcriber can add free comments:

[12] An end point for the alignment is not strictly required - it can be implied by simply taking the next start point of an IT-chunk in the same line. However, for some purposes (see, for instance, figure 20), specifiying an end point can supply additional information for the visualisation.

[13] In the EXMARaLDA API, for instance, possible attribute names (i.e. values of the 'property-name' attribute) are "font:name", "font:size", "font:face", "font:color", "bg:color", "chunk-border", "chunk-border:color" and "chunk-border:style". Other attribute names are ignored.

MAX	I don't know.	I think I'll go home now.	Good bye.
[ger]	*Ich weiß es nicht.*	*Ich glaube, ich gehe jetzt nach Hause.*	*Tschüß.*
[com]	This utterance is difficult to understand because somebody is fiddling with the microphone. Possibly, this may also be "I think I'm going home, you know."		

Figure 23

- Hyperlinks / Multimedia

Especially where IT is used to visualise transcriptions of spoken language, it may be useful to link parts of the textual description with corresponding sections of the original audio or video-recording. If the formal description of IT provides a place for such links, these can, for instance be "rendered" as hyperlinks in an HTML document, and the browser plug-ins can be used to play the files. See the EXMARaLDA homepage for some examples.

3.3. An API for IT

Currently, a large number of tools for working with linguistic annotation are developed at different sites[14]. Diverse as these tools may be, they all concentrate on the input and editing process for the data. The result of that process is usually a (XML-coded) file describing the logical structure of the data in a way that is suited for further automatic processing. What most of these tools do not provide, however, is a visualisation of the data optimised for further "processing by a human", i.e. a human-readable version of the machine-readable data. In spite of the fact that all these tools produce data that fulfils the requirements for being visualised as IT, none of them (with the exception of the TASX annotator) provides an IT output functionality.

It is often argued that, since the data are stored in XML, XSL transformations could be used to easily generate such visualisations in HTML or another presentation format. This process usually consists of two steps: in the first step, the logical representation of the data is transformed into some (formal) description of the visualisation, e.g. into one or several <table> or <p> element(s) in an HTML document, and in the second step, the visualisation is rendered on a screen or on a print-out by a program that "understands" the description, e.g. an internet browser:

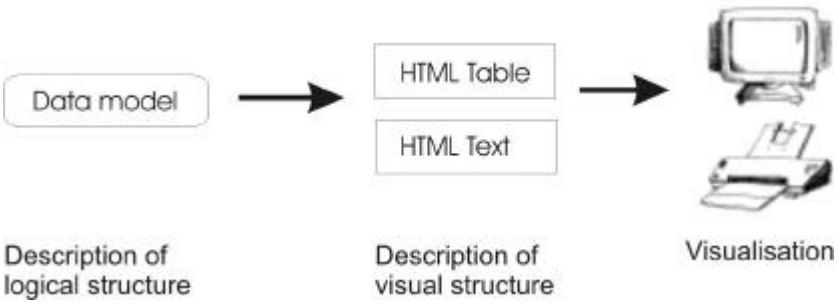

Figure 24

[14] The AG toolkit (Bird et al 2002), ELAN (http://www.mpi.nl/tools/elan.html), the TASX annotator (Milde/Gut 2001), ANVIL (http://www.dfki.de/~kipp/anvil) and Transcriber (Barras et al. 2000), just to name a few.

However, this process is not easily applied to IT. Applications like internet browsers and text processors do not "know" about IT and therefore neither provide ways for describing nor methods for rendering it. Elements like tables, tabulators etc. can be used to "imitate" the behaviour of IT, but this is by no means a straightforward process. Especially problematic is the fact that for a large number of rendering methods, IT has to be broken up in order to fit on a given page size. In linear text, this breaking is usually not part of the description of the visual structure, but something that is done programmatically by the application that renders the text. Any writer of a description of the visual structure (whether a machine or a human) can therefore restrict himself to the crucial aspects of this structure without having to bother with the (technically complex) breaking process.
In order to get a similarly comfortable way for working with IT, I would therefore suggest an API that takes care of this process, i.e. that takes a formal description of an IT (as described above) as an input and calculates an appropriate ("appropriate" also meaning "broken into appropriately sized pieces, if necessary") imitation of the description in a language that is understood by a rendering application:

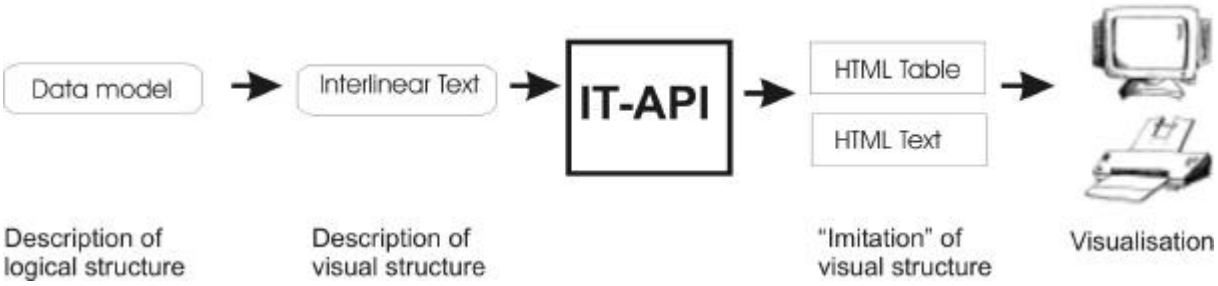

Figure 25

A prototype of such an API has been implemented in JAVA as a part of the EXMARaLDA system, and I will briefly outline its functionality in the next two sections.

3.3.1. Rendering IT

The formal description of IT suggested above defines *relative* positions of IT-Chunks by associating chunks on the same horizontal position with the same sync-points. However, in order to actually render such a description on a screen or a print-out, an *absolute* coordinate has to be calculated for each of these relative positions.

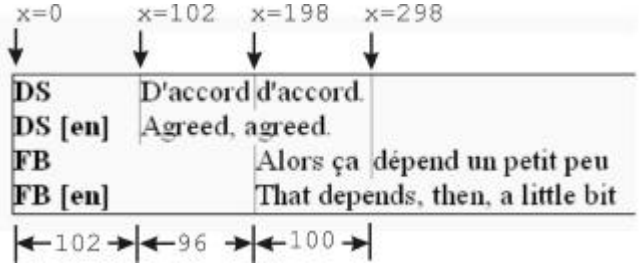

Figure 26

As the figure illustrates, this absolute horizontal position of aligned IT-chunks is a function of their typographical extent, i.e. it can be calculated only with the aid of the metrics of the underlying font(s)[15]. Once it is calculated, this information can be used to construct an

[15] This alone makes it very difficult to use XSLT, because XSLT does not provide any means for calculating the width of a given string in a given font. The actual calculations are mathematically quite complex, involving linear optimisation and hence the use of the simplex algorithm.

"imitation" of the IT in a table, with tabulators or other means that allow an absolute positioning of text (the widths of the columns of such a table or the position of the tab stops then correspond to the calculated values; making the tables "blind", i.e. without any visual cell borders, has the desired effect of making it look like IT):

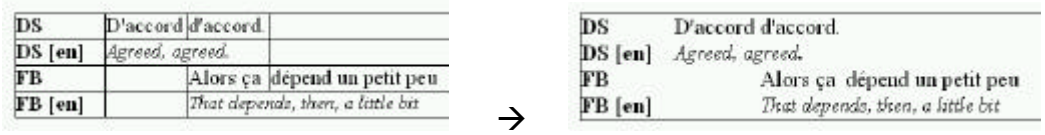

Figure 27

There are quite a number of potential target formats for the rendering. The EXMARaLDA prototype uses HTML and Microsoft's Rich Text Format because these two seemed to have the greatest practical use (HTML documents can easily be exchanged via the internet and RTF documents can be further processed in WORD). Other candidates would be PDF, LATEX or – once there are applications able to render it – XML formatting objects. In addition to that, the API has a print functionality making it possible to send an IT directly to the printer.

3.3.2. Breaking IT

Breaking IT up into several IT-Bundles that fit on a given page width is a recursive process, also done on the basis of font metrics calculations. The API calculates the last sync point that fully fits into the specified page width and cuts all IT-chunks aligned at that sync point to an appropriate size. The result is one IT-Bundle that fits on the page, and a second IT-Bundle onto which the same process can be applied:

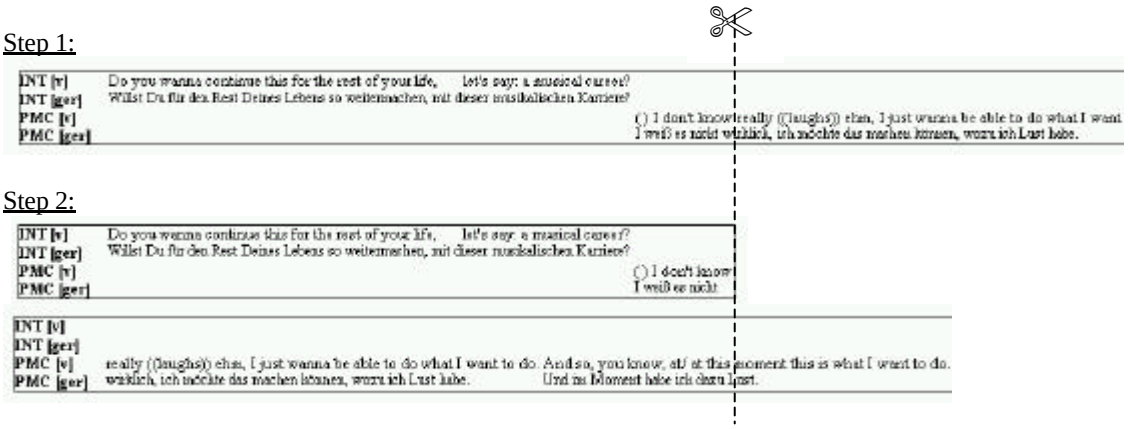

Figure 28

After the breaking process, IT bundles are often numbered for better orientation in the document, and empty IT lines (like the 'INT' lines in the second IT bundle in step 2 of the above example) are removed in order to save space. The API provides the functionality for this, too.

4. Application

The IT API, as outlined above, is currently used as a part of the EXMARaLDA system (Schmidt 2001). Building on the AG formalism, EXMARaLDA defines a "basic

transcription" format that meets the requirements for IT defined above, i.e. a basic transcription has one fully ordered timeline and distributes events onto several layers. Such a basic transcription can therefore be edited in the EXMARaLDA Partitur Editor, a GUI tool that presents a basic transcription as an (unbroken) interlinear text in a table:

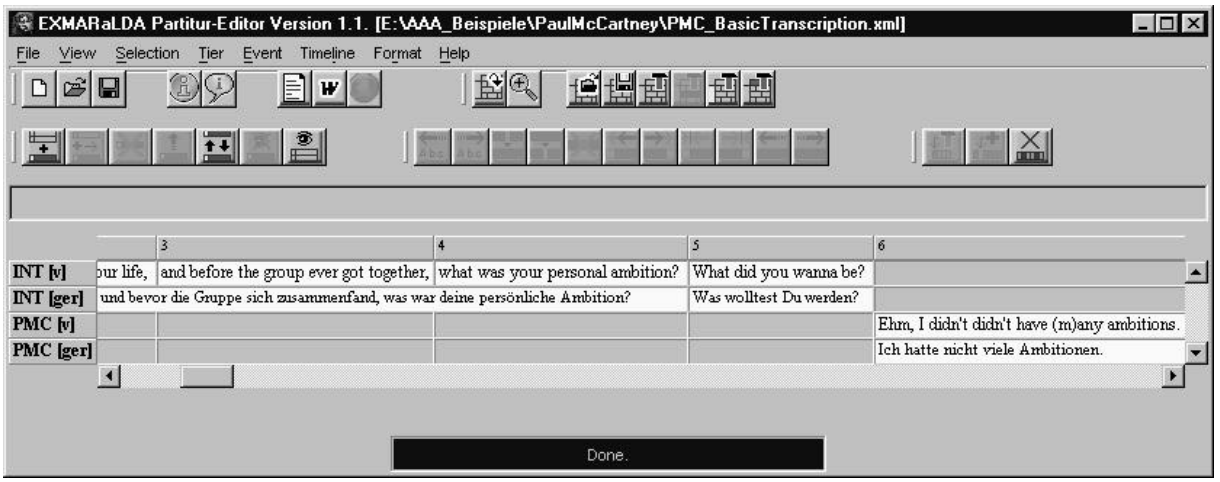

Figure 29

For output on a printer or as an RTF or HTML document, the basic transcription is first transformed into an IT. The API described above then takes care of breaking and rendering this IT in the desired manner. The appropriate parameters can also be set within the Partitur Editor:

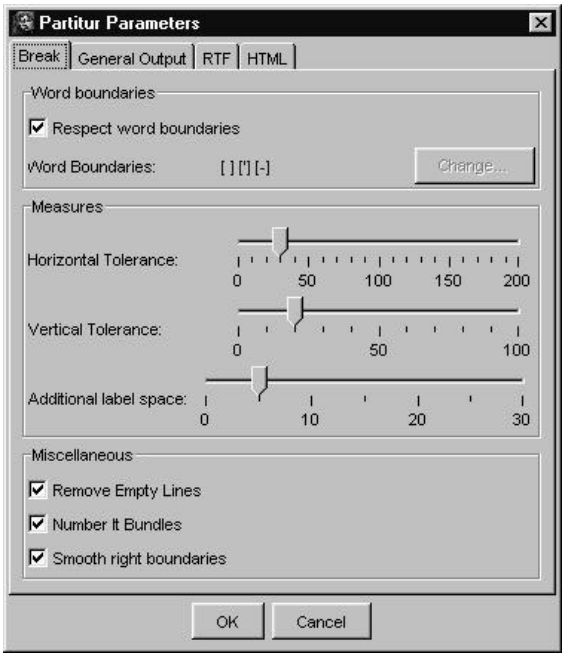

Figure 30

The EXMARaLDA basic transcription is, however, by no means the only data format that is suitable for a visualisation as IT. In fact, the large majority of transcription tools currently under development produces data that fulfil the requirements for such a visualisation. Version 1.1.1. of the EXMARaLDA Partitur Editor has import (and export) filters for TASX (Milde/Gut 2001) and Praat (http://www.praat.org) data, thus making it possible for such data

to be visualised as IT. This would be feasible for data from other tools (ELAN, ANVIL or the AG toolkit, for instance), too.

Conclusion

In this paper, I have shown that Interlinear Text is a widely used method in linguistics to visualise temporal and equivalence relations between units of linguistic models. I have argued that it is therefore best viewed as a visualisation method for data models rather than a data model in its own right. With the formal means of the annotation graph framework, it is possible to determine the conditions that a linguistic model must meet in order to have a visualisation as IT. In order to really work with IT in a comfortable and flexible manner, I have suggested a formal description of IT in the form of an XML document type definition, and an API that is able to manipulate these kinds of documents and prepare them for rendering on screen or paper. Finally, I have shown how this API is put to use as a part of the EXMARaLDA system.

References

Balthasar, Lucas (2001): *Transcrire les interactions filmées. Système de Transcription Audio-Visuelle des Interactions Sociales (STAVIS). Interaction audio-visuelle, théorie pragma-linguistique et transcription, vol. 2.* Thèse de Doctorat de l'Ecole des Hautes Etudes en Sciences Sociales. Paris.

Barras, Claude / Geoffrois, Edouard / Wu, Zhibiao / Liberman, Mark (2000): *Transcriber: development and use of a tool for assisting speech corpora production.* In: Speech Communication 33 (1,2), 5-22.

Bird, Steven / Liberman, Mark (2001): *A formal framework for linguistic annotation.* In: Speech Communication 33(1,2), pp. 23-60.

Bird, Steven / Buneman, Peter / Liberman, Mark (Hrsg.) (2001): *Proceedings of the IRCS Workshop On Linguistic Databases, 11-13 December 2001. Institute for Research in Cognitive Science. Philadelphia: University of Pennsylvania.*

Bird, Steven / Maeda, Kazuaki / Ma, Xiaoyi / Lee, Haejoong / Randall, Beth / Zayat, Salim (2002): *TableTrans, MultiTrans, InterTrans and TreeTrans: Diverse Tools Built on the Annotation Graph Toolkit.*

Du Bois, John / Schuetze-Coburn, Stephan / Cumming, Susanne / Paolino, Danae (1992): *Outline of Discourse Transcription.* In: Edwards/Lampert (1992).

Dybkjaer, Laila / Berman, Stephen / Kipp, Michael / Olsen, Malene / Pirelli, Vito / Reithinger, Norbert / Soria, Claudia (2001): *Survey of Existing Tools, Standards and User Needs for Annotation of Natural Interaction and Multimodal Data.* ISLE Natural Interactivity and Multimodality Working Group Deliverable D11.1.

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

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.

Ehlich, Konrad / Rehbein, Jochen (1976): *Halbinterpretative Arbeitstranskriptionen (HIAT).* In: Linguistische Berichte 45, 21-41.

Ehlich, Konrad / Rehbein, Jochen (1979a): *Zur Notierung nonverbaler Kommunikation für diskursanalytische Zwecke (HIAT2).* In: Winkler, Peter (Hrsg.) Methoden der Analyse von Face-To-Face-Situationen. Stuttgart, 302-329.

Ehlich, Konrad / Rehbein, Jochen (1979b): *Erweiterte halbinterpretative Arbeitstranskriptionen (HIAT 2): Intonation.* In: Linguistische Berichte (59), 51-75.

Ervin-Tripp, Susan M. (1979): *Children's verbal turn-taking.* In: Ochs/E. / Schieffelin, B. (Hrsg.): Developmental pragmatics. New York: Academic. 391-414.

Henne, Hartmut / Rehbock, Helmut (2001): *Einführung in die Gesprächsanalyse.* Berlin: de Gruyter.

Jacobson, Michel / Michailovsky, Boyd / Lowe, John B. (2000): *Linguistic documents synchronizing sound and text.* In: Speech Communication 33 (1,2), 79-96.

Klein, Wolfgang / Schütte, Wolfgang (2000): *Transkriptionsrichtlinien für die Eingabe in DIDA.* Mannheim: Institut für deutsche Sprache.

MacWhinney, Brian (2000): *The CHILDES project : tools for analyzing talk.* Mahwah, NJ: Lawrence Erlbaum (2 volumes).

Maeda, Kazuaki / Bird, Steven (2000): *A Formal Framework For Interlinear Text.* Paper presented at the workshop on Web-Based Language Documentation and Description, Philadelphia, USA.

Milde, Jan-Torsten / Gut, Ulrike (2001): *The TASX-Environment: an XML-based corpus database for time aligned language data.* In: Bird / Liberman / Buneman (2001), 174-180.

Rehbein, Jochen et al. (1992): *Manual für das computergestützte Transkribieren mit dem Programm syncWRITER nach dem Verfahren der Halbinterpretativen Arbeitstranskriptionen (HIAT).* Hamburg.

Schmidt, Thomas (2001): *The transcription system EXMARaLDA: An application of the annotation graph formalism as the Basis of a Database of Multilingual Spoken Discourse.* In: In: Bird / Liberman / Buneman (2001), 219-227.

Selting, Margret et al. (1998): *Gesprächsanalytisches Transkriptionssystem (GAT).* In: Linguistische Berichte 173, 91-122.

Sprouse, Ronald (2000): *Data types for interlinear text.* Paper presented at the workshop on Web-Based Language Documentation and Description, Philadelphia, USA.

Tannen, Deborah (1984): *Conversational style.* Norwood, NJ: Ablex.

Walter, Eric (1990): *Definition und Implementation eines Editors für Texte mit interlinearer Struktur.* Studienarbeit am Fachbereich Informatik der Universität Hamburg.

Wohlberg, Tim (1999): *Hypertables: Entwicklung einer Strukturbeschreibungssprache für Tabellen in XML.* Diplomarbeit am Fachbereich Informatik der Universität Hamburg.