FOLKER-Datenmodell

Thomas Schmidt, 15. Dezember 2008
Änderungen: 05. Februar 2009, 06. März 2009

FOLKER-Datenmodell

1. Erläuterungen 3
2. Grundlegende Struktureinheiten ("Gerüst") 4
    Folker-Transkription <folker-transcription> 4
    Kopf <head> 4
    Sprecherliste <speakers> 4
    Sprecher <speaker> 5
    Aufnahme <recording> 5
    Zeitachse <timeline> 5
    Zeitpunkt <timepoint> 5
    Beiträge <contribution> 6
3. Transkriptstufen ("Parse-Levels") 7
4. Parse-Level 0 8
    Segmente 8
5. Parse-Level 1 9
    Ungeparster Text <unparsed> 9
    Zeitverweis <time> 9
6. Parse-Level 2: Minimaltranskription 10
  6.1. Selbstständige Einheiten ("Grundbausteine") 10
    Wörter <w> 10
    Pausen <pause> 10
    Nichtphonologisches <non-phonological> 11
    Atmen <breathe> 11
    Unverständliches <unintelligible> 10
  6.2. Unsichere und Alternativ-Lautung 12
    Unsichere Lautung <uncertain> 12
    Alternativlautung <alternative> 12
7. Parse-Level 3: Basistranskription 13
  7.1. Phrasen 13
  7.2. Kommentare 13
  7.3. Nicht-Selbstständige Einheiten 13
    Akzent <stress> 13
    Dehnung <lengthening> 13
8. Feintranskription 14

1. Erläuterungen

Dieses Dokument beschreibt das FOLKER-Datenmodell, auf dem das von FOLKER geschriebene und gelesene XML-Datenformat basiert. Das Dokument ist zu lesen als Ergänzung zum XML-Schema "Folker_Schema.xsd", das die hier erläuterten Elemente in einer formalen Dokumentgrammatik beschreibt. Allerdings werden hier teilweise zusätzliche strukturelle Anforderungen an die Daten beschrieben, die von dem Schema nicht abgedeckt sind und bei einer Schema-Validierung nicht überprüft werden. Dies betrifft insbesondere allerlei Anforderungen zur zeitlichen Struktur.

Dieses Dokument richtet sich an Entwickler, die das Folker-Datenformat verstehen und verarbeiten wollen. Es werden folgende typographischen Konventionen verwendet:

Einheiten der Transkription werden in gelb hinterlegten Kästen eingeführt. In diesen Kästen steht links bündig der deutscher Name der Einheit, rechts das entsprechende XML-Element.

Folker-Transkription	**<folker-transcription>**

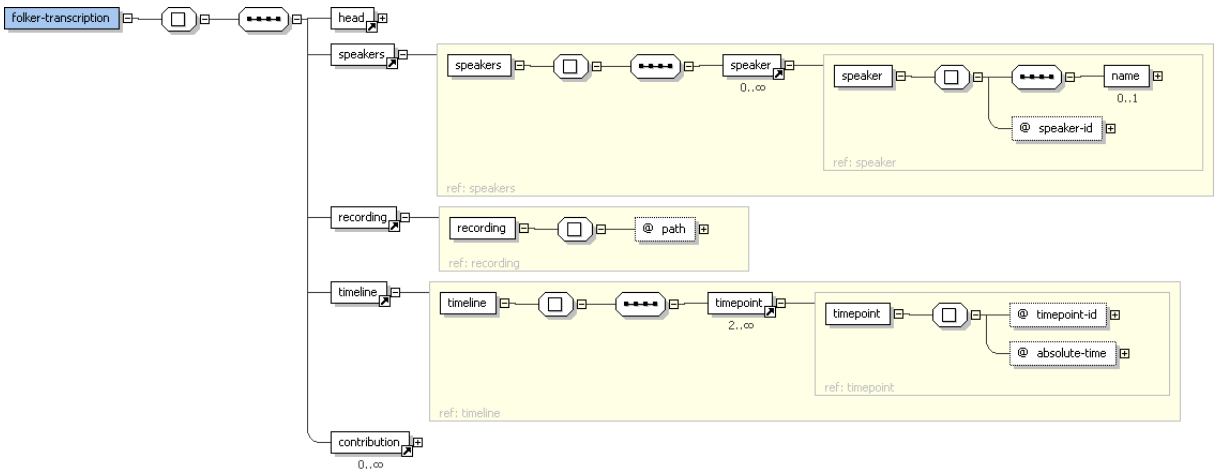
Eigenschaften von Einheiten der Transkription werden in blau hinterlegten Kästen eingeführt. In diesen Kästen steht links bündig der deutscher Name der Eigenschaft, rechts das entsprechende XML-Attribut.

Sprecher-ID	**@speaker-id**

Wenn innerhalb des Textes auf eine Einheit, eine Eigenschaft, ein XML-Element oder ein XML-Attribut Bezug genommen wird, so ist dieses grau hinterlegt.

Die Einheit Folker-Transkription umfasst die Gesamtheit aller Transkriptionsdaten

2. Grundlegende Struktureinheiten ("Gerüst")



Folker-Transkription **<folker-transcription>**

Die Einheit Folker-Transkription umfasst die Gesamtheit aller Transkriptionsdaten und enthält in dieser Reihenfolge:

- genau einen Kopf
- genau eine Sprecherliste
- genau eine Aufnahme
- genau eine Zeitachse
- eine beliebig lange (auch leere) Liste von Beiträgen

Dabei ist die Liste der Beiträge nach deren zeitlicher Reihenfolge geordnet (siehe dazu unter Beitrag).

Kopf **<head>**

Der Kopf ist z.Z. leer [no pun intended]. Dies wäre aber potentiell der Ort, um

1) einen vordefinierten GAT-Transkriptkopf (Transkribent, Name, etc.) oder
2) eine Liste frei definierbarer Attribut-Wertpaare oder
3) einen vordefinierten FOLK-Transkriptkopf oder
4) Metadaten gemäß einem vorhandenen Metadatenvokabular (z.B. OLAC, Dublin Core) oder
5) eine Mischung aus 1-4

zu definieren.

Sprecherliste **<speakers>**

Die Einheit Sprecherliste enthält eine beliebig lange (auch leere) Liste von Sprechern. Für diese Liste ist keine feste Ordnung definiert – Sprecher werden i.d.R. einfach in der Reihenfolge geführt, in der sie in den Daten angelegt wurden.

Sprecher **<speaker>**

Sprecher-ID **@speaker-id**

Jeder Sprecher ist eindeutig durch eine obligatorische Sprecher-ID identifiziert. Diese ID muss mit einem der Buchstaben *A-Z* oder *a-z* beginnen, dem eine im Prinzip beliebig lange Folge weiterer Buchstaben oder Ziffern (aber: keine Leerzeichen, keine Interpunktion etc.) folgen darf. Die ID muss im Bezug auf alle IDs in der Transkription eindeutig sein, d.h. es dürfen weder zwei Sprecher die gleiche Sprecher-ID haben, noch darf eine Sprecher-ID identisch mit einer Zeitpunkt-ID sein.

Aufnahme **<recording>**

Pfad **@path**

Jeder Transkription muss genau eine Aufnahme zugeordnet werden. Die Aufnahme wird über ihren Pfad identifiziert. Der Pfad kann ein relativer sein (z.B. 'Audio/Aufnahme17.wav') und wird dann mit Bezug auf den Speicherort der Datei interpretiert.

Zeitachse **<timeline>**

Die Einheit Zeitachse enthält eine Liste von zwei oder mehr Zeitpunkten . Die Zeitpunkte müssen dabei aufsteigend nach ihren absoluten Zeitangaben geordnet sein. Es dürfen in der Zeitachse weder zwei Zeitpunkte mit identischen absoluten Zeitangaben vorkommen, noch Zeitpunkte, deren Zeitangabe über die Länge der Aufnahme hinausgeht.

Zeitpunkt **<timepoint>**

Zeitpunkt-ID **@timepoint-id**

absolute Zeitangabe **@absolute-time**

Jeder Zeitpunkt ist eindeutig durch eine obligatorische Zeitpunkt-ID identifiziert. Diese ID muss mit einem der Buchstaben *A-Z* oder *a-z* beginnen, dem eine im Prinzip beliebig lange Folge weiterer Buchstaben oder Ziffern (aber: keine Leerzeichen, keine Interpunktion etc.) folgen darf. Die ID muss im Bezug auf alle IDs in der Transkription eindeutig sein, d.h. es dürfen weder zwei Zeitpunkte die gleiche Zeitpunkt-ID haben, noch darf eine Zeitpunkt-ID identisch mit einer Sprecher-ID sein.
Für jeden Zeitpunkt ist eine absolute Zeitangabe in Form einer positiven Fließkommazahl (also z.B. *1.45* oder *2* oder *.3*) obligatorisch. Diese wird als Zeitangabe in Sekunden interpretiert.

Beiträge	<contribution>

Sprecherzuordnung	@speaker-reference
Start-Zuordnung, End-Zuordnung	@start-reference, @end-reference
Transkriptstufe	@parse-level

Beiträge strukturieren das transkribierte Material. Jeder Beitrag ist obligatorisch mit zwei zeitlichen Zuordnungen versehen – der Start-Zuordnung und der End-Zuordnung. Diese verweisen über Zeitpunkt-IDs auf Zeitpunkte, die betreffenden Zeitpunkte müssen also in der Zeitachse vorhanden sein. Weiterhin muss der Zeitpunkt, auf den die Start-Zuordnung verweist, in der Zeitachse **vor** dem Zeitpunkt liegen, auf den die End-Zuordnung verweist (identische Werte sind also nicht erlaubt).

Jeder Beitrag kann (muss aber nicht) außerdem eine Sprecherzuordnung erhalten, indem er über eine in der Sprecherliste vorkommende Sprecher-ID auf einen Sprecher verweist.

Die Ordnung der Beiträge ergibt sich aus ihren Start- und Endzuordnungen. Bei unterschiedlichen Startzuordnungen erscheint derjenige Beitrag zuerst, der die **frühere** Startzuordnung hat. Bei identischen Startzuordnungen erscheint derjenige Beitrag zuerst, der die **spätere** Endzuordnung hat. Bei gleichzeitig beginnenden Beiträgen wird also der längere Beitrag vor den kürzeren gesetzt. Sind sowohl Start- als auch Endzuordnung zweier Beiträge gleich, ist deren Ordnung beliebig.

Wie ein Beitrag in sich strukturiert ist, hängt von der Transkriptstufe (s.u.) ab:

- auf Level 0 besteht ein Beitrag aus einer Liste von einem oder mehr Segmenten,
- auf Level 1 besteht ein Beitrag aus einem ungeparstem Text,
- auf Level 2 (Minimaltranskription) besteht ein Beitrag aus einer Folge von Grundbausteinen, zwischen denen Zeitverweise platziert sein können,
- auf Level 3 (Basistranskription) besteht ein Beitrag aus einer Liste von einer oder mehr Phrasen.

Die Transkriptstufe kann, muss aber nicht, für jeden Beitrag individuell in einem Attribut in Form einer Zahl zwischen 0 und 3 festgehalten werden.

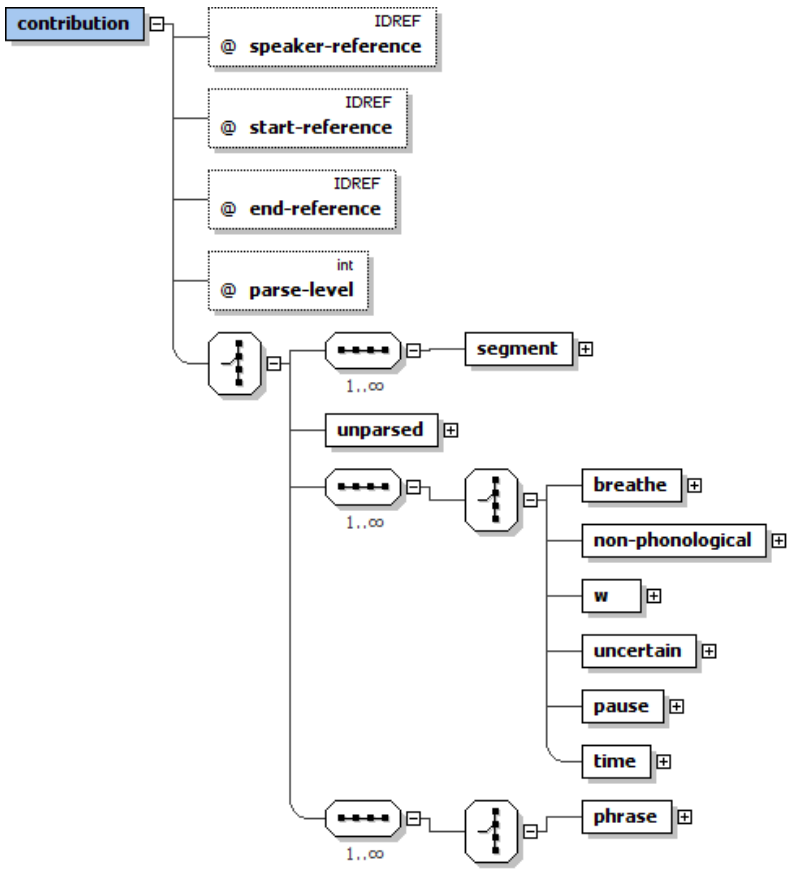
Für alle zeitlichen Verweise innerhalb eines Beitrags (aus Zeitverweisen oder in Form der Start- und Endzuordnung eines Segments) muss gelten, dass sie sich nicht auf Zeitpunkte außerhalb des Intervalls, das durch die Start- und Endzuordnung des Beitrags selbst definiert ist, beziehen dürfen.

3. Transkriptstufen ("Parse-Levels")

Jeder Beitrag in einer Transkription kann nach aufeinander aufbauenden Transkriptstufen geparst werden. Dabei werden folgende Stufen unterschieden:

- Level 0: Es erfolgen weder Checks der zeitlichen Integrität (d.h. insbesondere, dass es innerhalb von Beiträgen Selbstüberlappungen geben kann) noch wird die Syntax des Transkriptionstextes überprüft.
- Level 1: Es wird überprüft, ob einzelne Beiträge vollständig zeitlich geordnet sind, d.h. ob sie aus einer zeitlich zusammenhängenden Folge von Segmenten bestehen.
- Level 2 (Minimaltranskription): Beiträge werden nach Grundbausteinen geparst, d.h. es wird überprüft, ob die Syntax für Wörter, Pausen, Nicht-Phonologisches, Schwerverständliches, Alternativlautungen und Ein-/Ausatmen korrekt angewendet wurde.
- Level 3 (Basistranskription): Beiträge werden nach Phrasen geparst.

Um sicherzustellen, dass auch eine nicht vollständig korrekte Transkription jederzeit in ein wohl definiertes Dateiformat geschrieben werden kann, erfolgt das Parsing grundsätzlich beitragsweise. Scheitert das Parsing eines Beitrags auf Transkriptstufe n+1, so wird dieser Beitrag im Dateiformat auf der Transkriptstufe n gespeichert. Die vollständige Transkription ist genau dann erfolgreich auf Transkriptstufe n geparst, wenn jeder einzelne Beitrag auf Transkriptionsstufe n geparst werden konnte.



4. Parse-Level 0

Segmente ****

Zeitliche Zuordnung **@start-reference, @end-reference**

Beim Erstellen von Transkriptionen werden zunächst nicht Beiträge, sondern Segmente erstellt. Ein Segment ist eine beliebige textuelle Beschreibung, die einem beliebigen zeitlichen Abschnitt der Aufnahme und optional einem Sprecher zugeordnet ist. Die zeitliche Zuordnung erfolgt per Zeitpunkt-IDs auf Zeitpunkte, die betreffenden Zeitpunkte müssen also in der Zeitachse vorhanden sein. Dabei muss der Startzeitpunkt in der Zeitachse vor dem Endzeitpunkt liegen.
Aus der Gesamtmenge der Segmente werden dann Beiträge wie folgt errechnet:

1) Für jeden Sprecher wird eine Liste von Segmenten erstellt, die diesem Sprecher zugeordnet sind.
2) Die Liste wird nach den Anfangspunkten der Segmente geordnet.
3) Die Liste wird in der vorgegebenen Reihenfolge durchlaufen.
4) Für das erste Segment wird ein neuer Beitrag angelegt.
5) Für Segment n+1 gilt: wenn der Anfangspunkt dieses Segments vor dem Endpunkt von Segment n liegt oder mit diesem identisch ist, wird das Segment dem aktuellen Beitrag hinzugefügt. Wenn das nicht der Fall ist, wird ein neuer Beitrag angelegt und das Segment diesem neuen Beitrag hinzugefügt.
6) Jedes Segment, das keinem Sprecher zugeordnet sind, wird einem eigenen Beitrag zugeordnet (d.h. sprecherlose Beiträge enthalten grundsätzlich genau ein Segment).

Auf Parse-Level o wird genau diese Struktur erzeugt und gespeichert. Es gilt immer, dass der Startpunkt des ersten Segments mit dem Startpunkt des übergeordneten Beitrags identisch ist, ebenso der Endpunkt des letzten Segments mit dem Endpunkt des übergeordneten Beitrags. Die Sprecherzuordnung einzelner Segmente wird nur im übergeordneten Beitrag festgehalten.

```
<timeline>
   <timepoint timepoint-id="TLI_18" absolute-time="14.2"/>
   <timepoint timepoint-id="TLI_19" absolute-time="16.1"/>
   <timepoint timepoint-id="TLI_20" absolute-time="19.3"/>
   <timepoint timepoint-id="TLI_21" absolute-time="20.4"/>
</timeline>
<!-- Segment 2 folgt unmittelbar auf  Segment 1 -->
<contribution speaker-reference="R" start-reference="TLI_18" end-reference="TLI_20">
   <segment start-reference="TLI_18" end-reference="TLI_19">segment1</segment>
   <segment start-reference="TLI_19" end-reference="TLI_20">segment2</segment>
</contribution>
<!-- Segment 2 beginnt vor dem Ende von Segment 1 -->
<contribution speaker-reference="F" start-reference="TLI_18" end-reference="TLI_20">
   <segment start-reference="TLI_18" end-reference="TLI_21">segment1</segment>
   <segment start-reference="TLI_19" end-reference="TLI_20">segment2</segment>
</contribution>
<!-- Segmente 1 und 2 sind keinem Sprecher zugeordnet -->
<contribution start-reference="TLI_18" end-reference="TLI_20">
   <segment start-reference="TLI_18" end-reference="TLI_21">segment1</segment>
</contribution>
<contribution start-reference="TLI_19" end-reference="TLI_20">
   <segment start-reference="TLI_19" end-reference="TLI_20">segment2</segment>
</contribution>
```

5. Parse-Level 1

Ungeparster Text	<unparsed>

Zeitverweis	<time>

Zeitliche Zuordnung	@timepoint-reference

Auf Parse-Level 1 wird für jeden Beitrag geprüft, ob die darin enthaltenen Segmente eine zeitlich zusammenhängende Folge bilden. Notwendige und hinreichende Voraussetzung dafür ist, dass der Startpunkt von Segment n+1 identisch mit dem Endpunkt von Segment n ist. Ist dies der Fall, wird statt einer expliziten Folge von segment-Elementen ein Element unparsed erzeugt, das

1) den Text der Ausgangssegmente in deren zeitlicher Reihenfolge und
2) an jeder Segmentgrenze, die zwischen Start- und Endpunkt des übergeordneten Beitrags liegt, einen Zeitverweis auf den entsprechenden Zeitpunkt enthält.

Wenn ein Beitrag zeitlich überlappende Segmente enthält, wird die Repräsentation aus Parse-Level 0 beibehalten, d.h. Segmente werden explizit kodiert.

```
<timeline>
   <timepoint timepoint-id="TLI_18" absolute-time="14.2"/>
   <timepoint timepoint-id="TLI_19" absolute-time="16.1"/>
   <timepoint timepoint-id="TLI_20" absolute-time="19.3"/>
   <timepoint timepoint-id="TLI_21" absolute-time="20.4"/>
</timeline>
<!-- Dieser Beitrag enthält keine Überlappungen -->
<contribution speaker-reference="R" start-reference="TLI_18" end-reference="TLI_20">
   <unparsed>segment1<time timepoint-reference="TLI_19"/>segment2</segment></unparsed>
</contribution>
<!-- Dieser Beitrag enthält Überlappungen -->
<contribution speaker-reference="F" start-reference="TLI_18" end-reference="TLI_20">
   <segment start-reference="TLI_18" end-reference="TLI_21">segment1</segment>
   <segment start-reference="TLI_19" end-reference="TLI_20">segment2</segment>
</contribution>
<!-- Segmente 1 und 2 sind keinem Sprecher zugeordnet -->
<contribution start-reference="TLI_18" end-reference="TLI_20">
   <unparsed>segment1</unparsed>
</contribution>
<contribution start-reference="TLI_19" end-reference="TLI_20">
   <unparsed>segment2</unparsed>
</contribution>
```

6. Parse-Level 2: Minimaltranskription

Auf Parse-Level 2 wird versucht, innerhalb von Beiträgen, die Parse-Level 1 passiert haben, die Grundbausteine der Transkription sowie unsicher identifzierte Lautungen und ggf. zugehörige Vorschläge für Alternativlautungen zu identifizieren. Jeder Grundbaustein folgt dabei einem vorgegebenen Zeichenmuster.

6.1. Selbstständige Einheiten ("Grundbausteine")

Wörter **<w>**

Übergang **@transition**

Ein Wort besteht aus einer beliebigen Folge von Kleinbuchstaben des deutschen Alphabets (also die Buchstaben a-z, ä. ö. ü und ß). Wortgrenzen werden beim Transkribieren durch ein Leerzeichen oder, im Falle einer Assimilation, durch einen Unterstrich repräsentiert, im Datenmodell selbst tauchen sie nicht auf. Stattdessen wird ggf. für das folgende Wort vermerkt, dass der Übergang vom vorhergehenden eine Assimilation ist (d.h. das Attribute @transition erhält den Wert assimilated; wenn das Attribut @transition nicht vorhanden ist, handelt es sich um eine nicht übersprochene Wortgrenze).

Zulässige Zeichenmuster	Unzulässige Zeichenmuster	
schön	Schön	[Großbuchstabe]
schön gesagt	geSAGT	[Großbuchstaben]
schön gesagt aber schlecht gedacht	citroën	[unzulässiges Zeichen]
was_n		
mi_m hammer		
wi_r_i		

Unverständliche Wörter **<w>**

ÄNDERUNG, 06.03.2009

Gänzlich Unverständliches, bei dem sich die Silbenzahl bestimmen lässt, wird als Wort transkribiert, bei dem für jede Silbe die Zeichenkette +++ steht. Ggf., also wenn Wortgrenzen vermutet werden, können zwischen diesen Zeichenketten auch Leerzeichen gelassen werden – es entstehen dann mehrere solcher Wörter.[1]

Zulässige Zeichenmuster	Unzulässige Zeichenmuster	
+++	++	[kein Vielfaches von 3]
+++++++++	+++++++	[dito]
+++ ++++++ +++		

[1] Dass hier genau drei Zeichen gefordert werden, ist eigentlich eine willkürliche Festlegung, führt aber dazu, dass unverständliche Silben optisch in etwa so viel Raum einnehmen wie eine transkribierte Silbe. Es bietet sich an, das gleiche Verfahren auch zur Anonymisierung zu verwenden. Dabei können die Leerzeichen gemäß Wortgrenzen gesetzt werden. Sehr lange unverständlichen Passagen oder solche, bei denen auch die ungefähre Silbenzahl nicht ermittelt werden kann, können auch als Nichtphonologisches mit einer geeigneten Beschreibung (z.B. "((unverständlich, 5s))") repräsentiert werden.

Pausen <pause>

Dauer @duration

Es gibt geschätzte und gemessene Pausen. Bei den gemessenen Pausen werden vier Typen – Mikropause, kurze Pause, mittellange Pause und lange Pause – unterschieden. Die Werte für deren Dauer werden entsprechend als micro, short, medium bzw. long angegeben. Die Dauer einer gemessenen Pause wird in Sekunden als positive Fließkommazahl mit einer oder zwei Nachkommastellen angegeben. Als Dezimalpunkt wird der Punkt (nicht das Komma!) verwendet, als Höchstdauer werden 99.99 Sekunden angenommen.

Zulässige Zeichenmuster		Unzulässige Zeichenmuster	
(.)	[Mikropause]	(----)	[zu viele Bindestriche]
(-)	[kurze Pause]	(0,5)	[Komma statt Punkt]
(--)	[mittellange Pause]	(0.563)	[zu viele Nachkommastellen]
(---)	[lange Pause]	(121.4)	[zu große Zahl]
(1.23)	[gemessene Pause]		
(0.3)	[gemessene Pause]		
(34.88)	[gemessene Pause]		

Nichtphonologisches <non-phonological>

Beschreibung @description

Nichtphonologisches (z.B. Husten, Räuspern, Lachen und andere akustisch wahrnehmbare Handlungen, die nicht parallel zu gesprochenem verlaufen) wird generell durch eine freie Beschreibung repräsentiert. Das Zeichenmuster besteht aus einem Paar öffnender und schließender runder Klammern, zwischen denen die Beschreibung steht. Die Beschreibung darf Großbuchstaben, Zahlen, Interpunktion etc. enthalten, sie darf allerdings keine weitere öffnende Klammer enthalten und nicht mit einem Punkt, einem Bindestrich oder einer Ziffer beginnen.

Zulässige Zeichenmuster		Unzulässige Zeichenmuster	
((niest)		((1 Sekunde Husten))	[Ziffer am Anfang]
((niest, schneuzt sich))	[Interpunktion]	((--- Kratzen ---))	[Bindestrich am Anfang]
((startet seinen VW))	[Großschreibung]	((zwei (oder drei?) Sekunden Applaus))	
((startet seinen Citroën))	[zusätzliche Zeichen]		[zusätzliche Klammern]

Atmen <breathe>

Typ @type

Dauer @length

Beim Atmen werden die beiden Typen Einatmen (@type='in') und Ausatmen (@type='out') unterschieden. Bei der Länge werden drei diskrete Stufen unterschieden, die mit den Zahlen von 1 bis 3 beschrieben werden. Das Textmuster besteht aus ein- bis

dreimaliger Wiederholung des Buchstabens 'h' (je nach Länge) und einem vorangestellten (Einatmen) bzw. nachfolgenden (Ausatmen) Gradzeichen.

Zulässige Zeichenmuster	Unzulässige Zeichenmuster	
°h	°hhhh	[zuviele 'h']
°hh	°°hhh	[zwei Gradzeichen]
°hhh		
h°		
hh°		
hhh°		

6.2. Unsichere und Alternativ-Lautung

Unsichere Lautung **<uncertain>**

Alternativlautung **<alternative>**

Passagen, bei denen der Transkribent sich über seine Transkription nicht sicher ist, können entsprechend markiert werden. Gegebenenfalls können zu einer unsicheren Lautung eine oder mehrere Alternativlautungen angegeben werden. Eine unsicher verstandene Passage kann ein oder mehrere Wörter umfassen, es ist allerdings nicht erlaubt, die Grenzen einer solchen Passage innerhalb eines Wortes zu setzen oder etwas anderes als (ggf. assimilierte) Wörter einer solchen Passage zuzuordnen. Das gleiche gilt für Alternativlautungen. Das Textmuster besteht aus einem Paar runder Klammern, die die gesamte Passage umfassen. Innerhalb der Passage werden Vorschläge für Alternativlautungen durch den Forward-Slash (Schrägstrich: '/') abgetrennt.

Zulässige Zeichenmuster	Unzulässige Zeichenmuster	
(unsicher)	(°hhh)	[Nichtwort]
(ganz unsicher)	(un)sicher	[Wortteil]
(ganz unsicher/franz hunziger)	(ganz unsicher/)	[leere Alternative]
(s_is/miss)		

7. Parse-Level 3: Basistranskription

7.1. Phrasen

7.2. Kommentare

7.3. Nicht-Selbstständige Einheiten

Akzent	**<stress>**
Dehnung	**<lengthening>**
Ausprägung	**@length**

8. Feintranskription

(=Einheiten, die in verschiedenen GAT-Versionen erwähnt werden, aber keinen Eingang in das Datenmodell finden sollen)

Knarrstimme, Akzentstärke (Haupt- vs. Nebenakzent), Tonhöhensprünge, Tonhöhenregister, Akzenttonhöhenbewegungen, Lautstärke- und Sprechgeschindigkeitsveränderungen