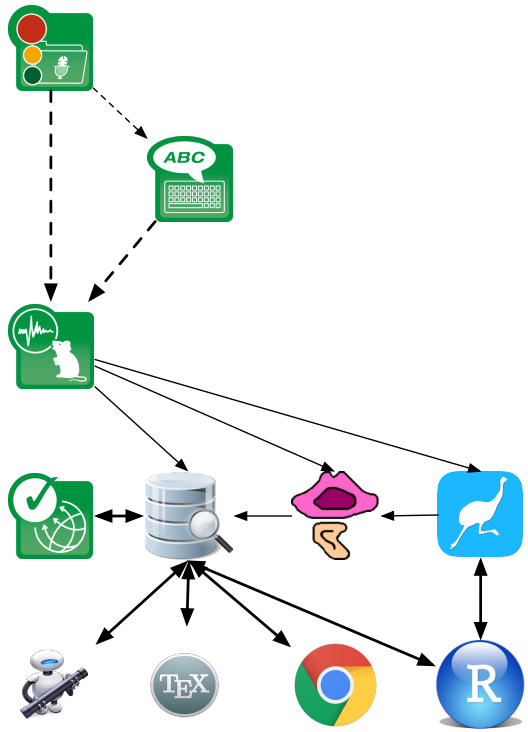


Erstellung von Sprachdatenbanken – Automatisierung des Workflows

Christoph Draxler
Bayerisches Archiv für Sprachsignale (BAS)
draxler@phonetik.uni-muenchen.de

zentrale Thesen

- Tools und Forscher/innen gehen, Daten bleiben
- in Daten und Abläufen denken, nicht in Tools!
- bestehende Lösungen immer wieder hinterfragen
- Informatiker einbeziehen

Beispiel: TIMIT Sprachdatenbank

Kooperation von Texas Instruments und MIT [GLF+86]

- 1986 auf CD-ROM veröffentlicht
- ursprünglich zur Entwicklung von Spracherkennern
- Signaldaten, Text und phonemische Segmentation in eig. Alphabet
- Hinzukommen *immer noch*
  - weitere Fragestellungen
  - weitere Annotationen
  - weitere Sprachen
  - weitere Kommunikationskanäle und -bedingungen

Die damals Beteiligten sind im Rentenalter, keiner der Rechner läuft mehr, keines der Tools ist noch verfügbar – aber die Daten leben und gedeihen!

Bayerisches Archiv für Sprachsignale: Dienste und Tools

G2P	Graphem-zu-Phonem-Konverter
SpeechRecorder	skriptgesteuerte Sprachaufnahmen
OCTRA	online Transkriptionseditor(en)
WebMAUS	automatische phonemische Segmentation (30 Sprachen bzw. Varietäten)
emu WebApp	online Annotationseditor und R-Integration
COALA	Erstellung CLARIN-kompatibler Metadaten für Korpora
Percy	online Perzeptionsexperimente

Bayerisches Archiv für Sprachsignale: Repository

- aktuell 44 Sprachdatenbanken in 4 Sprachen
  - 34 eigene, 10 externe
  - 39 frei für akademische Nutzung
- Metadaten frei verfügbar und durchsuchbar

Annotationen

stand-off	Annotation getrennt von annotierten (Roh-)Daten; Verbindung über Indizes und Verweise
Ebene, Tier	organisatorische Einheit, fasst Elemente eines Typs zusammen
Item	kleinstes Annotationselement
Schema	formale Definition der Annotation

Rohdaten sind prinzipiell unveränderlich, Annotationsdaten können verändert, erweitert, gelöscht werden.

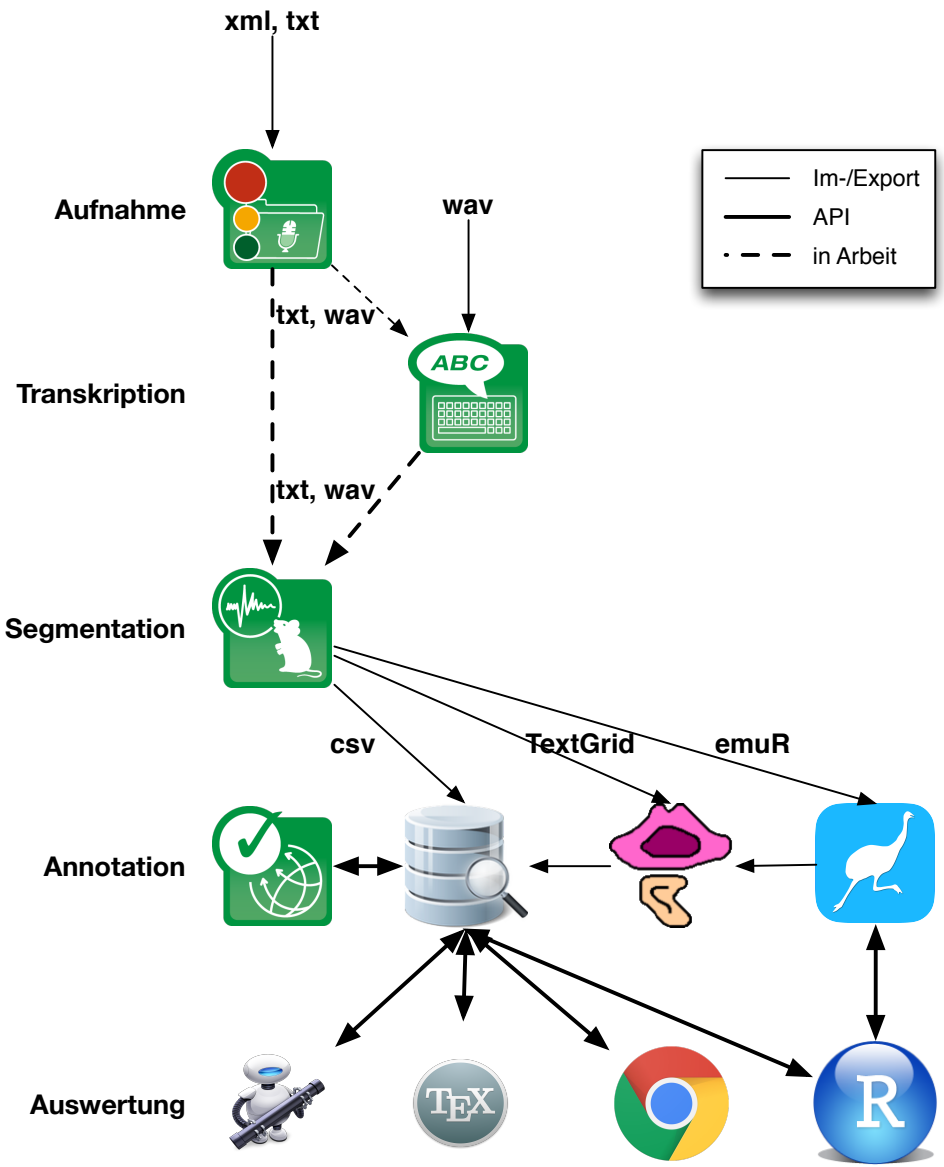
Annotation: Erfahrungswerte für Zeit und Kosten

[Kva93], [WMA+11] nennen folgende Werte

Aufgabe	Ebene	Kosten	Zeitfaktor
Chunking*	Schnittliste	€	2
Rohtranskript	Orthographie, Markup	€	10
kanonische Transkription	SAM-PA	€€	60
auditive Transkription	IPA	€€€	300
manuelle Segmentation	IPA, Zeitstempel	€€€	1200

Automatisierung besonders vielversprechend für Transkription und Segmentation

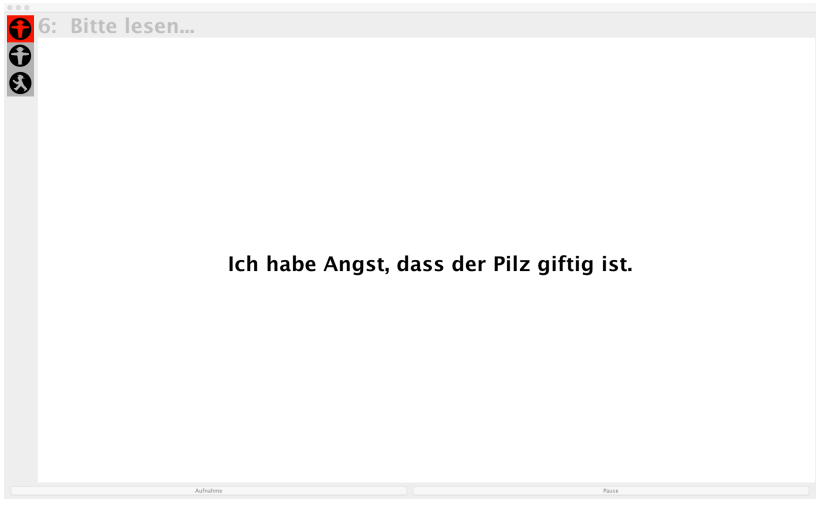
Arbeitsablauf, Tools und Daten



am BAS aktuell in Arbeit

- Export von Vorlagentext aus SpeechRecorder
- API-Aufrufe von WebMAUS aus SpeechRecorder bzw. OCTRA

Skriptgesteuerte Aufnahmen: SpeechRecorder



- Aufnahmeskript in XML
- Aufnahmeleiter- und Sprecheransicht
- erstellt `.wav` und optional `.txt` Dateien

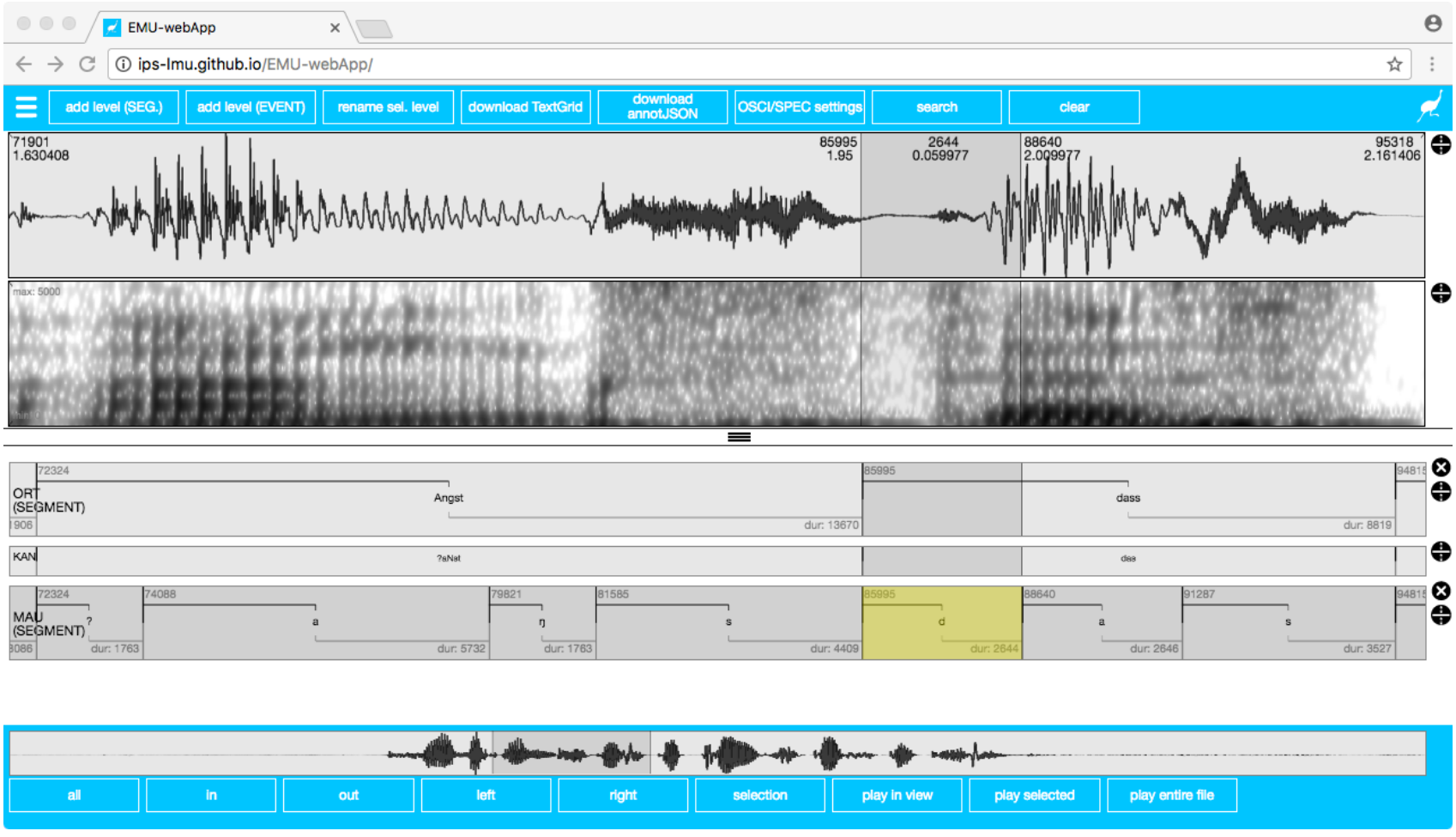
Orthographische Transkription: OCTRA

- lokaler und online-Modus
- `.wav`, `.txt` und `TextGrid`-Import
- schreibt Transkript in die Datenbank bzw. in eine emuR-Datei

Phonetische Segmentation

- emuR, `.wav` und `TextGrid`-Import
- schreibt Transkript in emuR-Textdatei

...Angst, dass...


EMU-webApp
ips-lmu.github.io/EMU-webApp/
add level (SEG.)
add level (EVENT)
rename sel. level
download TextGrid
download annotJSON
OSCI/SPEC settings
search
clear
71901
1.630408
85995
1.95
2644
0.059977
88640
2.009977
95318
2.161406
max: 5000
ORT (SEGMENT)
Angst
dass
KAN
MAU (SEGMENT)
all
in
out
left
right
selection
play in view
play selected
play entire file

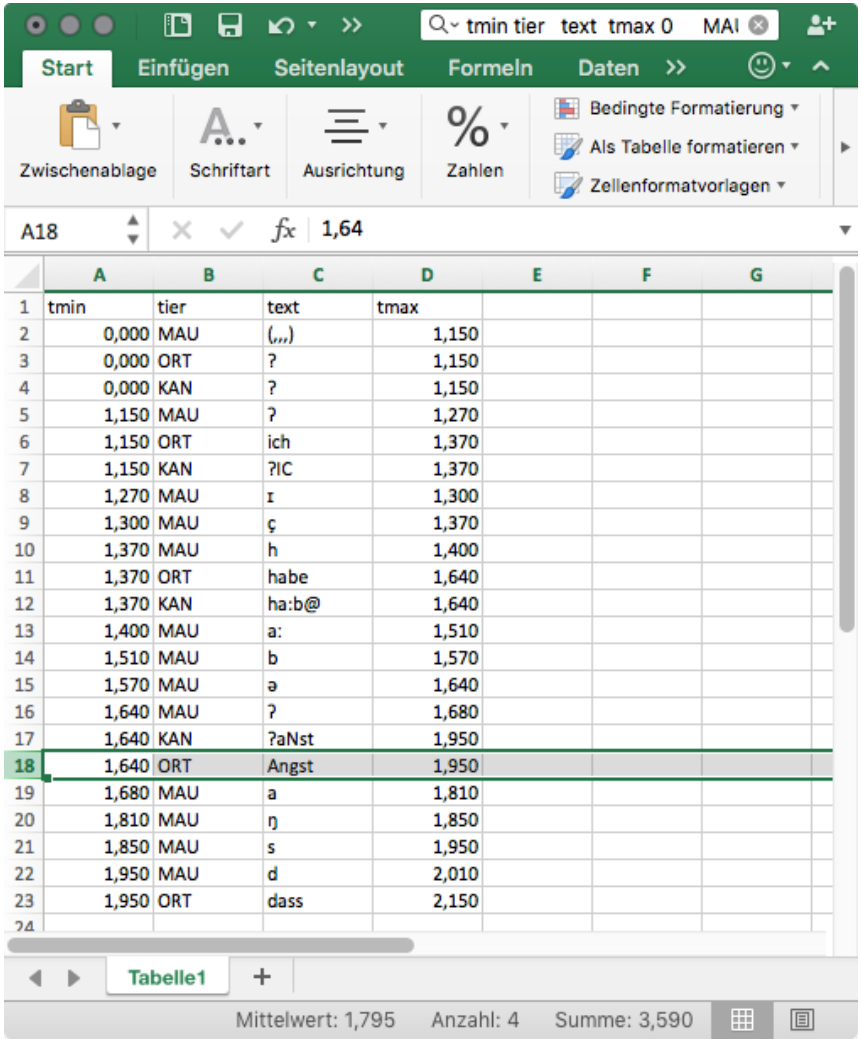
Praat TextGrid vs. Table

```
File type = "ooTextFile"
Object class = "TextGrid"

xmin = 0
xmax = 4.760000
tiers? <exists>
size = 3
item []:
    item [1]:
        class = "IntervalTier"
        name = "ORT"
        xmin = 0
        xmax = 4.760000
        intervals: size = 10
        intervals [1]:
            xmin = 0.000000
            xmax = 1.150000
            text = ""
        intervals [2]:
            xmin = 1.150000
            xmax = 1.370000
            text = "ich"
```

```
tmin tier text tmax
0 ORT ? 1.1500
0 KAN ? 1.1500
0 MAU (...) 1.1500
1.1500 ORT ich 1.3700
1.1500 KAN ?IC 1.3700
1.1500 MAU ? 1.2700
1.2700 MAU I 1.3000
1.3000 MAU ç 1.3700
1.3700 ORT habe 1.6400
1.3700 KAN ha:b@ 1.6400
1.3700 MAU h 1.4000
1.4000 MAU a: 1.5100
1.5100 MAU b 1.5700
1.5700 MAU @ 1.6400
1.6400 KAN ?aNst 1.9500
1.6400 ORT Angst 1.9500
1.6400 MAU ? 1.6800
1.6800 MAU a 1.8100
1.8100 MAU N 1.8500
1.8500 MAU s 1.9500
1.9500 ORT dass 2.1500
```

Excel bzw. relationale Datenbank



	A	B	C	D
1	tmin	tier	text	tmax
2	0,000	MAU	(,,,)	1,150
3	0,000	ORT	ʔ	1,150
4	0,000	KAN	ʔ	1,150
5	1,150	MAU	ʔ	1,270
6	1,150	ORT	ich	1,370
7	1,150	KAN	ʔIC	1,370
8	1,270	MAU	ɪ	1,300
9	1,300	MAU	ç	1,370
10	1,370	MAU	h	1,400
11	1,370	ORT	habe	1,640
12	1,370	KAN	ha:b@	1,640
13	1,400	MAU	a:	1,510
14	1,510	MAU	b	1,570
15	1,570	MAU	ə	1,640
16	1,640	MAU	ʔ	1,680
17	1,640	KAN	ʔaNst	1,950
18	1,640	ORT	Angst	1,950
19	1,680	MAU	a	1,810
20	1,810	MAU	ŋ	1,850
21	1,850	MAU	s	1,950
22	1,950	MAU	d	2,010
23	1,950	ORT	dass	2,150

```
speechdb=> select tier, position as pos, label,
  begin, duration from segment
  where signalfile_id = 16945177
  and tier in ('ORT', 'KAN', 'MAU')
  order by begin, tier, position;

 tier | pos |  label | begin | dur
------+-----+--------+-------+------
 MAU  |     | <p:>   | 27341 | 5292
 MAU  |   0 | h      | 32634 | 2204
 MAU  |   1 | a      | 34839 | 2424
 MAU  |   2 | b      | 37264 | 1542
 MAU  |   3 | @      | 38807 | 1102
 MAU  |   0 | a      | 39910 | 3306
 MAU  |   1 | N      | 43217 | 2425
 MAU  |   2 | s      | 45643 | 2424
 MAU  |   3 | t      | 48068 |  882
 MAU  |   0 | d      | 48951 |  881
 MAU  |   1 | a      | 49833 | 1542
 MAU  |   2 | s      | 51376 | 1322
 KAN  |   1 | h'ab@  |       |
 KAN  |   2 | ?'aNst |       |
 KAN  |   3 | d'as   |       |
 ORT  |   1 | habe   |       |
 ORT  |   2 | Angst  |       |
 ORT  |   3 | dass   |       |
...
```

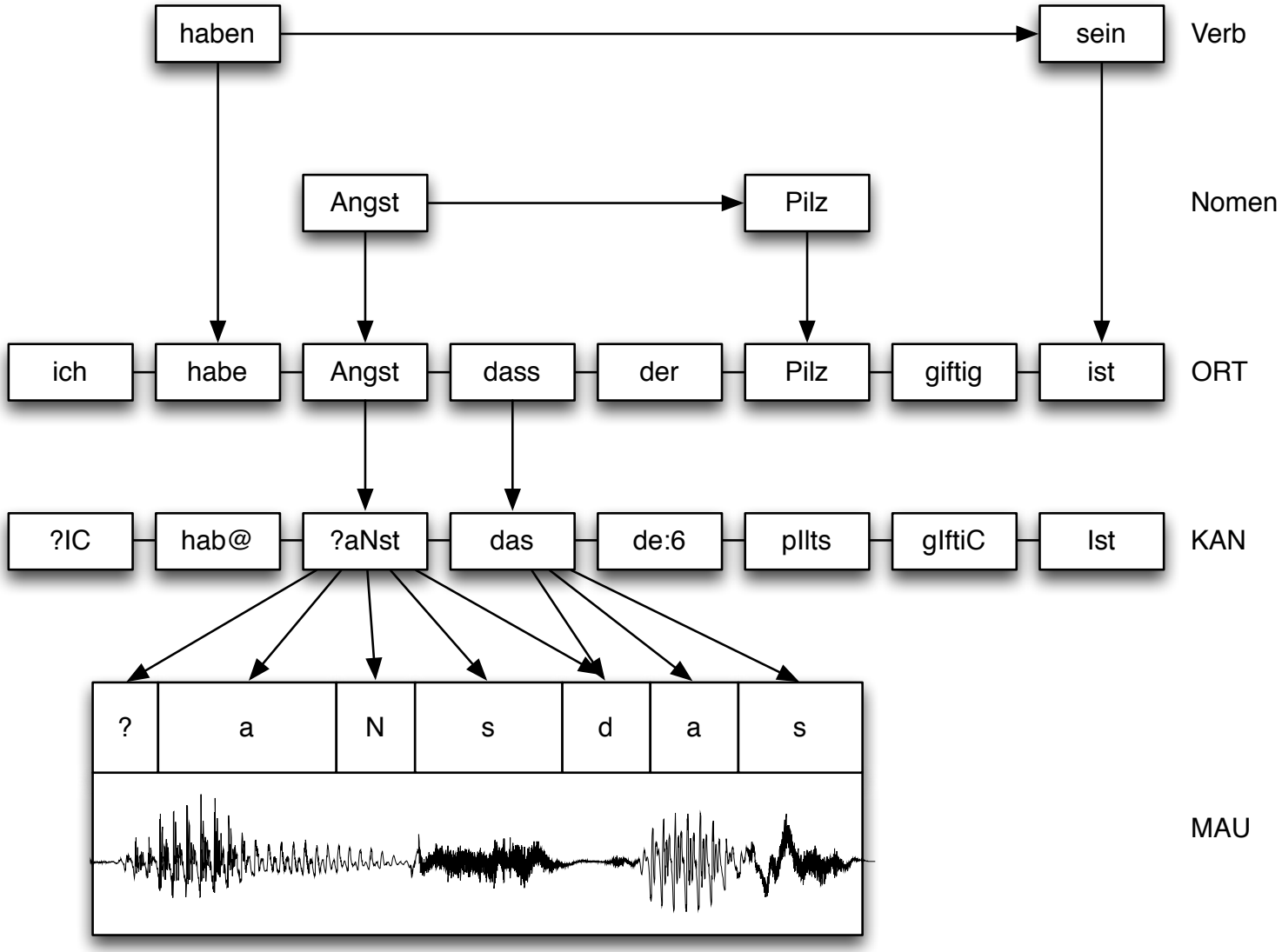
emu WebApp (JSON)


```
{ "name": "DRCH0036S1",
  "annotates": "DRCH0036S1.wav",
  "sampleRate": 44100,
  "levels": [
    {
      "name": "ORT",
      "type": "SEGMENT",
      "items": [
        {
          "id": 1,
          "sampleStart": 50714,
          "sampleDur": 9702,
          "labels": [{"name": "ORT", "value": "ich"}]
        },
        {
          "id": 2,
          "sampleStart": 60417,
          "sampleDur": 11906,
          "labels": [{"name": "ORT", "value": "habe"}]
        },
        {
          "id": 3,
          "sampleStart": 72324,
          "sampleDur": 13670,
          "labels": [{"name": "ORT", "value": "Angst"}]
        },
        {
          "id": 4,
          "sampleStart": 85995,
          "sampleDur": 8819,
          "labels": [{"name": "ORT", "value": "dass"}]
        },
        ...
```

Datenstrukturen in der Informatik

Objekt	gekapselte Datenstruktur mit Feldern, kann rekursiv sein
Sequenz	lineare, in der Regel nummerierte Folge gleichartiger Objekte
Graph	netzwerkartig organisierte Objekte, *Kanten* verknüpfen *Knoten*
Constraint	logische Bedingung(en) für Datenstrukturen

Beispiel


haben
sein
Verb
Angst
Pilz
Nomen
ich
habe
Angst
dass
der
Pilz
giftig
ist
ORT
?IC
hab@
?aNst
das
de:6
pIlts
gIftiC
Ist
KAN
?
a
N
s
d
a
s
MAU

Datenmodell für Annotationen I

Basisobjekte für die Annotation gesprochener Sprache sind

`Item`	Annotationslabel
`Event`	Item mit einem Zeitpunkt
`Segment`	Item mit zwei Zeitpunkten
`Link`	benannte Verknüpfung zweier Annotationsobjekte
`Constraints`	Abhängigkeiten zwischen Annotationsebenen

Ein Annotationsobjekt gehört immer einer *Annotationsebene* an.

Datenmodell für Annotationen II

Komplexe Annotationen bestehen aus vielen Annotationsobjekten

- Annotationsobjekte *mit* Zeitbezug sind innerhalb einer Ebene *implizit* sequenziell organisiert.
- Annotationsobjekte *ohne* Zeitbezug sind mit *Links* verknüpft.
- Links können *innerhalb* von Ebenen und *ebenenübergreifend* sein.
- Links müssen die Constraints des Schemas erfüllen.

Mit Annotationsebenen und Constraints geht diese Datenstruktur über die Annotation Graphs von [BL01] hinaus.

EmuR ist das erste (und meines Wissens nach das einzige) Tool, das dieses Datenmodell unterstützt.

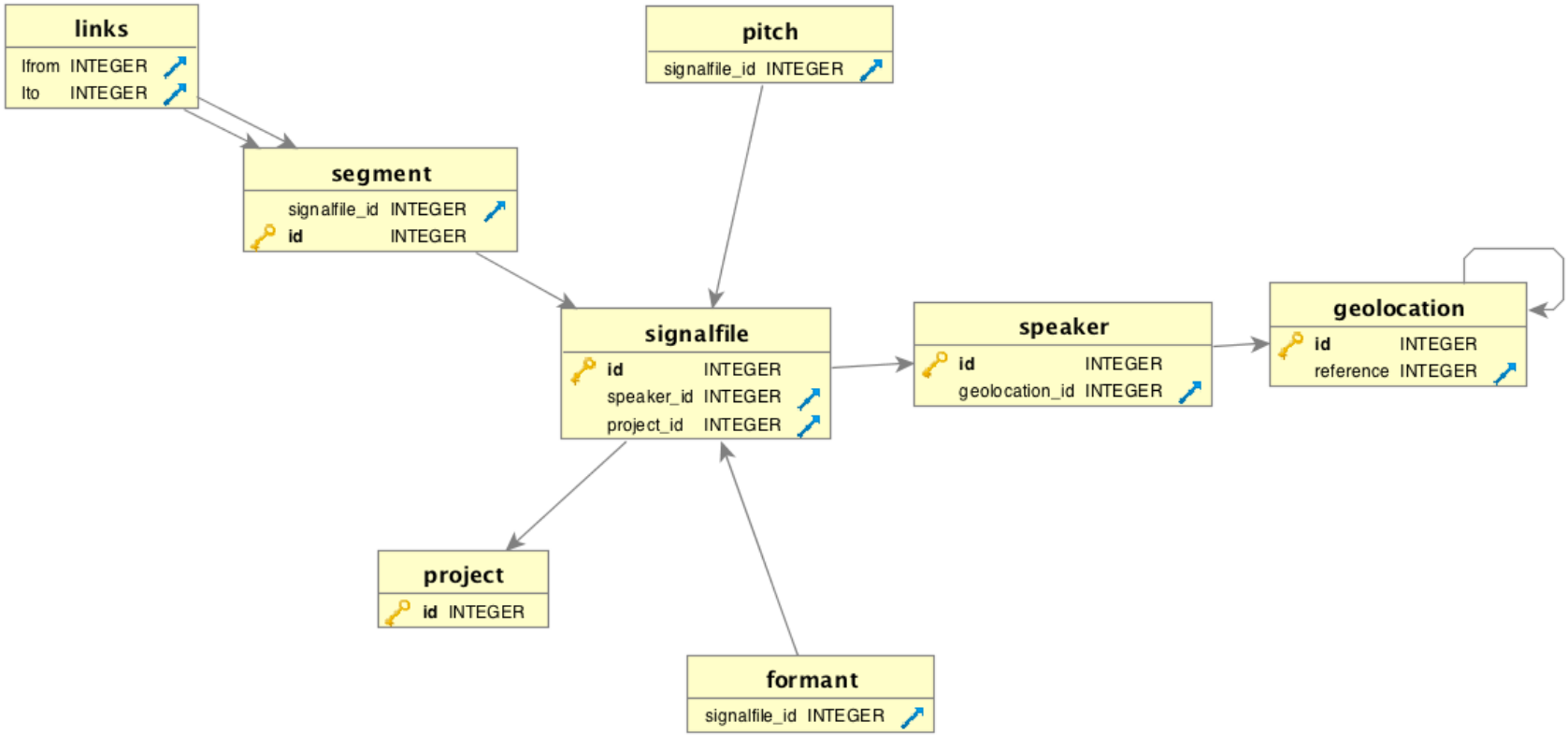
Empfehlungen

Den *gesamten* Arbeitsablauf und Datenfluss betrachten

- Datenfluss vollständig spezifizieren
- Tools auf ihr Datenmodell hin validieren
- Tool-unabhängige Datenspeicherung anstreben
- Arbeitsschritte wo immer möglich (teil-)automatisieren
- Annotationen immer durch ein Schema definieren und formal überprüfen
- Mindeststandard: UTF-8 Kodierung und plain text Tabellenformate, XML oder JSON

Informatiker und Fachwissenschaftler müssen hier kooperieren.

Beispiel: SpeechDB relationale Datenbank



konkret: PostgreSQL Datenbanksystem – Tool-unabhängig

Was kann die Datenbank?

Die Datenbank verwaltet

- Aufnahme- und Transkriptionsprojekte
- Sprecher
- Signaldateien und akustische Daten
- Ortsinformationen
- Vorlagentexte
- Basistranskripte und Segmentdaten

Sie bietet Mehrbenutzerbetrieb, zentrale Datenhaltung und eine standardisierte Schnittstelle für Tools und Anwendungsprogramme.

Was kann die Datenbank nicht?

Die Datenbank bietet keine

- Signalverarbeitung bzw. Audioausgabe
- Visualisierung oder mächtige Statistikfunktionen
- komfortable Nutzerschnittstelle
- anwendungsspezifische Abfragesprache

Diese Funktionalitäten müssen in externen Tools implementiert werden.

Akzeptanzprobleme

Ich höre häufig folgende Aussagen:

- Ich habe nicht die notwendige Informatikkompetenz für eine Datenbank!
- Ich will nicht von einem/r Datenbank Administrator/in abhängig sein!
- Ich fühle mich in meinem explorativen Arbeiten durch ein fixes Datenbankschema eingeschränkt!
- Ich will nicht noch eine weitere Programmiersprache lernen müssen!
- Ich habe nur wenige Daten, der Aufwand lohnt sich für mich nicht!

(Merkels und) meine Antwort: gemeinsam schaffen wir das!

Einfache SQL-Abfragen zu Sprechern

1. Suche alle männlichen Sprecher

   ```
   select * from speaker where sex = 'm';
   ```

2. Suche alle Sprecherinnen zwischen 18 und 21

   ```
   select * from speaker
   where sex = 'f'and age between 18 and 21;
   ```

3. Suche nur das Alter aller männlichen Sprecher, ohne Duplikate, und sortiere nach Alter

   ```
   select distinct age from speaker
   where sex = 'm'order by age;
   ```

Abfragen über mehrere Tabellen

1. Suche alle Orte in Bayern, für die es Sprecher gibt

```
select distinct geo.label, geo.iso3166_2
from geolocation geo
 join speaker spk on spk.geolocation_id = geo.id
where geo.iso3166_2 = 'DE-BY';
```

2. Suche alle Wörter der Äußerung in der Datei 'AAA4640X2_0' und sortiere sie nach Position

```
select ort.label, ort.position
from segment ort
 join signalfile sig on ort.signalfile_id = sig.id
where sig.filename = 'AAA4640X2_0'
 and ort.tier = 'ORT'
order by ort.position;
```

Komplexe Abfragen: ist bairisches 'acht' anders?

```
select case when geo.iso3166_2 = 'DE-BY' then 'BAV'
   else 'other' end as state, mau.label as phoneme, spk.sex,
   avg(mau.duration / (sig.samplerate * 0.001))::int as "duration (ms)",
   avg(f.f1)::int as f1, avg(f.f2)::int as f2
from signalfile sig
join segment ort on ort.signalfile_id = sig.id
join links l on ort.id = l.lto
join segment mau on mau.id = l.lfrom
join formant f on f.signalfile_id = mau.signalfile_id
join speaker spk on sig.speaker_id = spk.id
join geolocation geo on spk.geolocation_id = geo.id
join project pr on sig.project_id = pr.id
where spk.sex in ('f', 'm')
and ort.tier = 'ORT'
and mau.tier = 'MAU'
and ort.label = 'acht'
and mau.label = 'a'
and geo.iso3166_2 like 'DE-%'
and (sig.itemcode in ('01', '02', '03', '04', '05', '06', '07', '08',
      '09', '10', '11', '14', '23')
    and pr.name = 'PHATTSESSIONZ')
and f.sample between mau.begin and (mau.begin + mau.duration)
group by state, spk.sex, ort.label, mau.label
order by state, spk.sex, ort.label, mau.label;
```

Es geht auch kürzer: View definieren

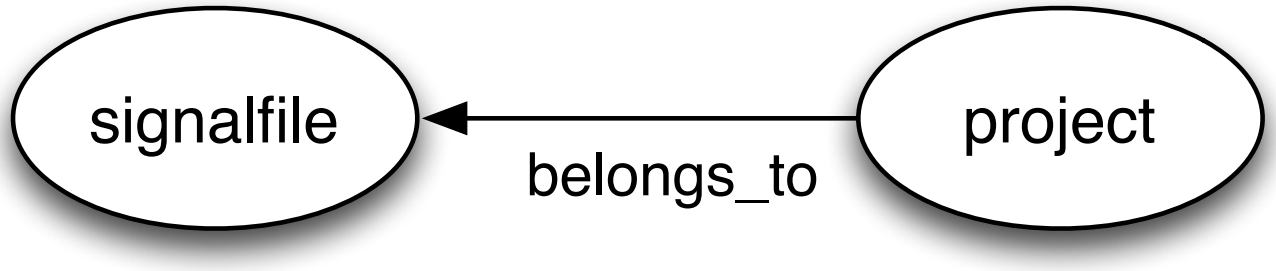
Eine *View* ist eine virtuelle Tabelle. Es lohnt sich, für häufig verwendete Abfragetypen eigene Views anzulegen. Damit lässt sich die Abfrage zum bairischen /a/ deutlich knapper formulieren:

```
select word, phoneme, sex, accent,
 avg(duration)::int, avg(f1)::int, avg(f2)::int
from formants_view
where word = 'acht' and phoneme = 'a'
group by word,phoneme, sex, accent
order by sex, avg(f1), avg(f2);
```

Fürwahr, eine ziemliche kurze Abfrage im Vergleich zu vorher!

Alternativen zu relationalen Datenbanken

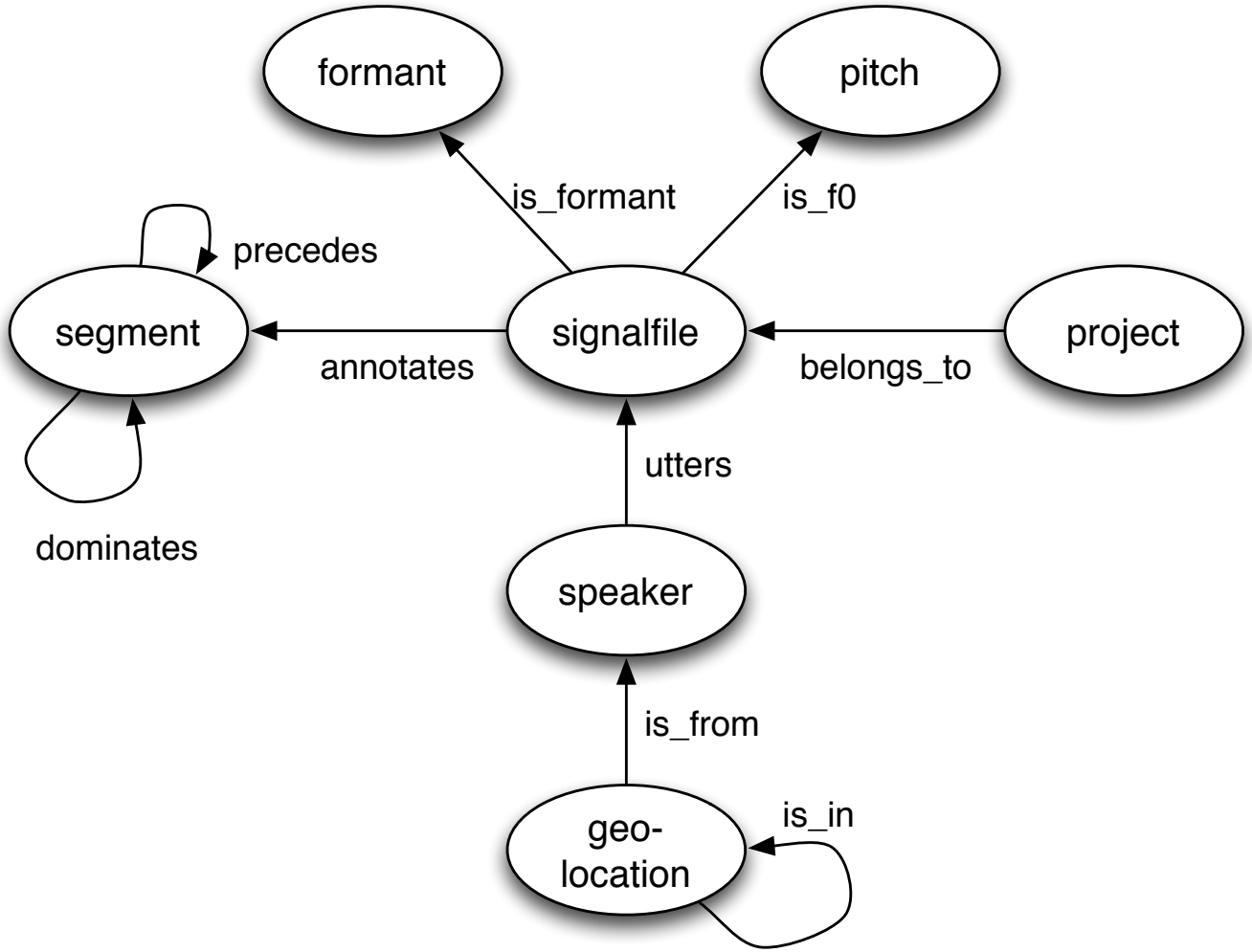
Graphen-Datenbanken sind eine relativ neue Entwicklung.



- Knoten sind durch gerichtete Kanten verbunden
- Knoten und Kanten sind benannt und haben Eigenschaften
- es gibt kein fixes Schema
- rekursive Abfragen sind erlaubt
- konzeptionell sehr nach an RDF

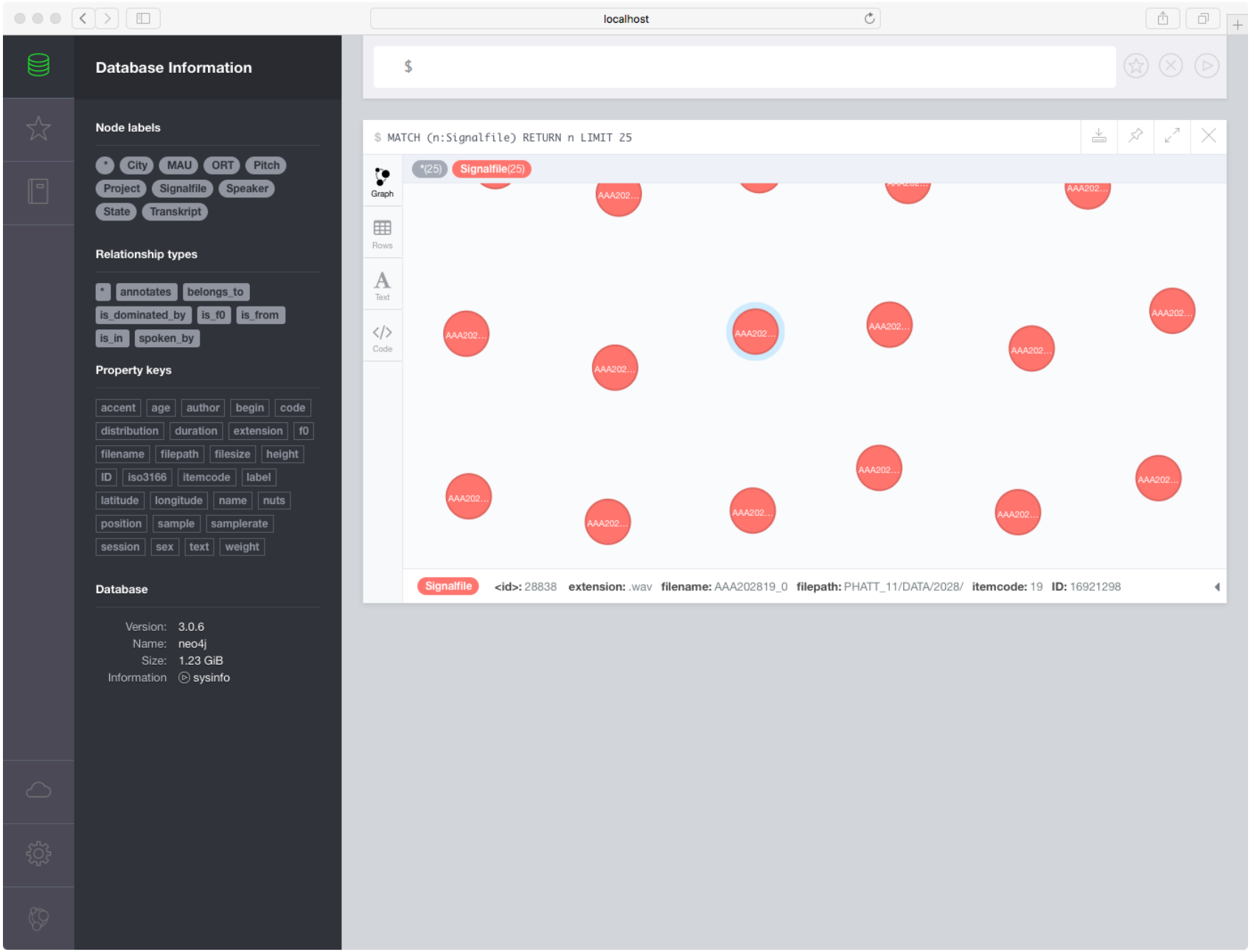
Ein Vertreter von Graphen-DBs ist *neo4j*.

Sprachdatenbank in Graphen-DB



Die Struktur der Datenbank ist etwas einfacher als die der relationalen Datenbank. Die benannten Kanten ersetzen die Fremdschlüssel und Hilfstabellen im relationalen Datenmodell.

Beispiel in neo4j



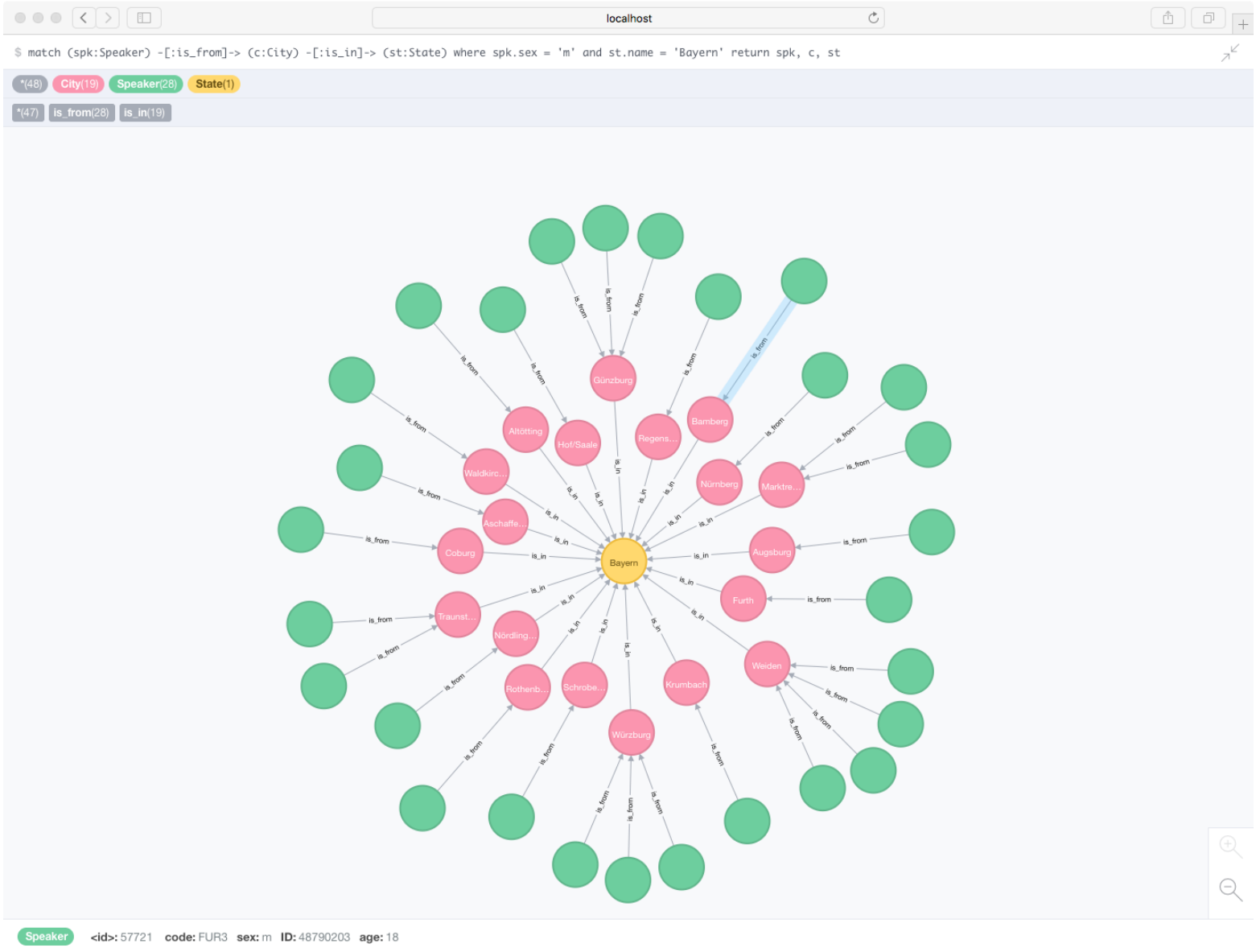
Cypher-Abfrage

Cypher ist eine deklarative SQL-ähnliche objekt-orientierte Abfragesprache

```
match (spk:Speaker) -[:is_from]-> (c:City)
  -[:is_in]-> (st:State)
where spk.sex = 'm' and st.name = 'Bayern'
return spk, c, st
```

gibt alle männlichen Sprecher aus Bayern zurück.

neo4j im Browser



Zusammenfassung

- Tools und Forscher/innen gehen, Daten bleiben
- in Daten und Abläufen denken, nicht in Tools!
- bestehende Lösungen immer wieder hinterfragen
- Informatiker einbeziehen
- Links
  - `https://clarin.phonetik.uni-muenchen.de/BASWebServices/`
  - `https://clarin.phonetik.uni-muenchen.de/BASRepository/`
  - `https://github.com/IPS-LMU/`


St. Bird and M. Liberman.
A Formal Framework for Linguistic Annotation.
Speech Communication, 33(1,2):23–60, 2001.


John Garofolo, Lori Lamel, William Fisher, Jonathan Fiscus, David S. Pallett, and Nancy Dahlgren.
The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus CDROM.
NIST, 1986.


K. Kvale.
Segmentation and Labelling of Speech.
PhD thesis, Norwegian Institute of Technology, Trondheim, 1993.


J. Williams, I. Melamed, T. Alonso, B. Hollister, and J. Wilpon.
Crowd-sourcing for difficult transcription of speech.
In *Proc. of IEEE Workshop on Automatic Speech Recognition and Unterstanding (ASRU 2011))*, 2011.