CLARIN-D

CLARIN Webservices
(CLARIN Repos)

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

Florian Schiel
Bavarian Archive for Speech Signals, Munich, Germany
hdl.handle.net/11858/00-1779-0000-000C-DAAF-B

BAS CLARIN Webservices

- webservice demos
- some basic concepts and definitions...
- structure of BAS webservices
- using the backend via REST calls
- the web interface (frontend)
- service metadata WADL / CMDI
- implementation details and logistics
- pro and cons

hdl.handle.net/11858/00-1779-0000-0028-421B-4

Demo 1 : text-to-phoneme

- typical problem:
  derive standard pronunciation from text
- input data: text file
- output data: text file with IPA
- Webinterface: G2P

- here: French sentence

hdl.handle.net/11858/00-1779-0000-0028-421B-4

Demo 2 : automatic segmentation

- typical problem:
  word/phonetic segmentation of
  speech signal
- input data: sound file + text file
- output data: word and phonetic
  segmentation in praat
- Webinterface: WebMAUS Basic

- here: Georgian recording

hdl.handle.net/11858/00-1779-0000-0028-421B-4

Demo 3 : word indexation of video

- typical problem:
  align script to video signal
- input data: sound track + text file
- output data: word and phonetic segmentation in praat
- Webinterface: WebMAUS Basic

- here: Goethe's Faust

hdl.handle.net/11858/00-1779-0000-0028-421B-4

Demo 3 : word indexation of video

hdl.handle.net/11858/00-1779-0000-0028-421B-4

Demo 4 : phonetic segmentation

- typical problem:
  segment and label speech signal into IPA segments
- input data: sound track
- output data: phonetic segmentation in praat
- Webinterface: WebMinni

- here: spontaneous Italian

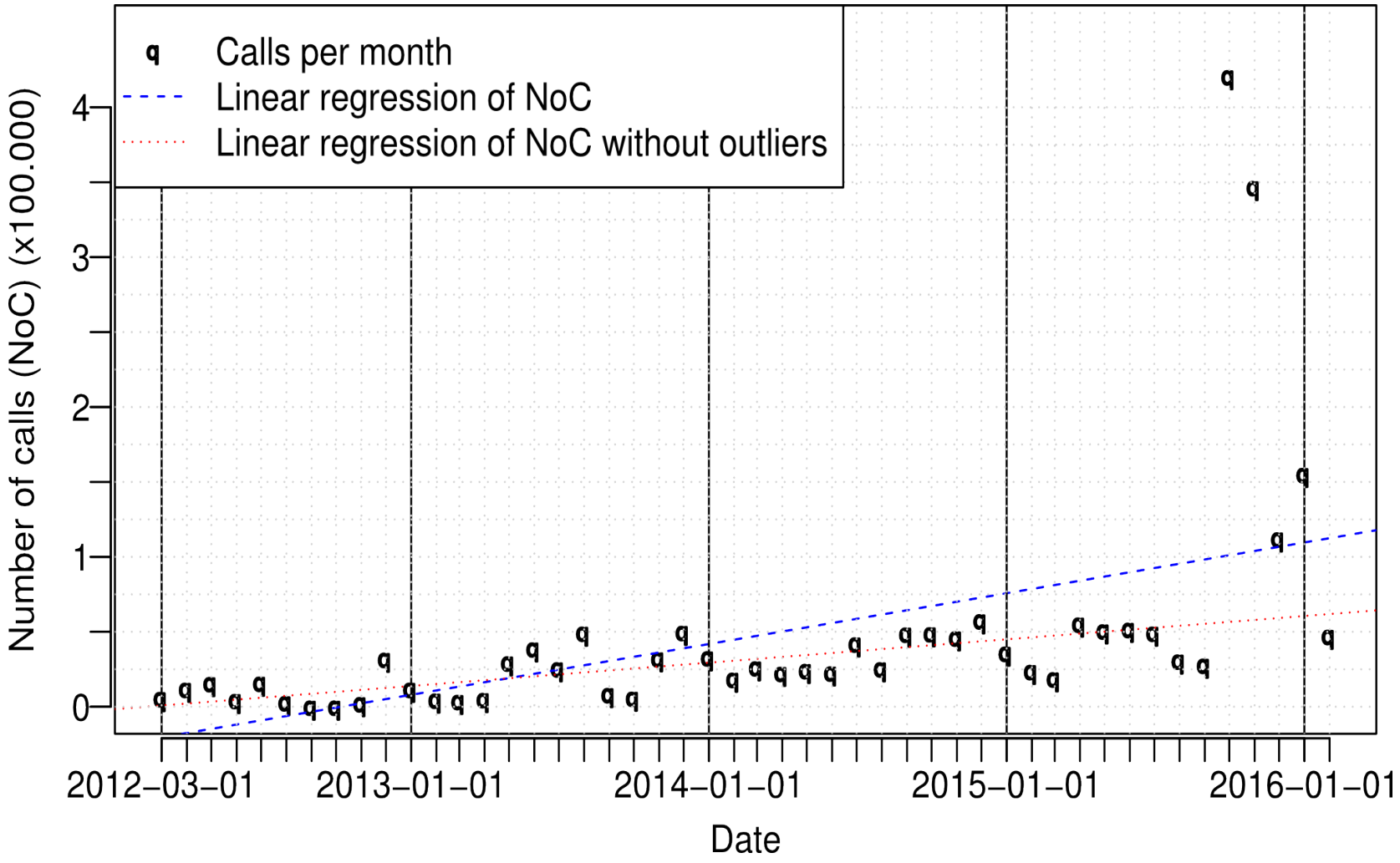
hdl.handle.net/11858/00-1779-0000-0028-421B-4

BAS Webservices: where to start?

- General Help with tutorials, FAQs, use cases, example files
- help button on individual service page
- hoover messages
- '?' buttons on parameters

hdl.handle.net/11858/00-1779-0000-0028-421B-4

BAS Webservices: Users?



hdl.handle.net/11858/00-1779-0000-0028-421B-4

Basic concepts, definitions

- 'webservice' :
  server based implementation of a tool,
  accessible via Internet,
  highly available,
  described by metadata (WADL,CMDI),
  non-interactive,
  called by applications (usually REST)

- 'webinterface' :
  front end to a webservice,
  user friendly,
  interactive

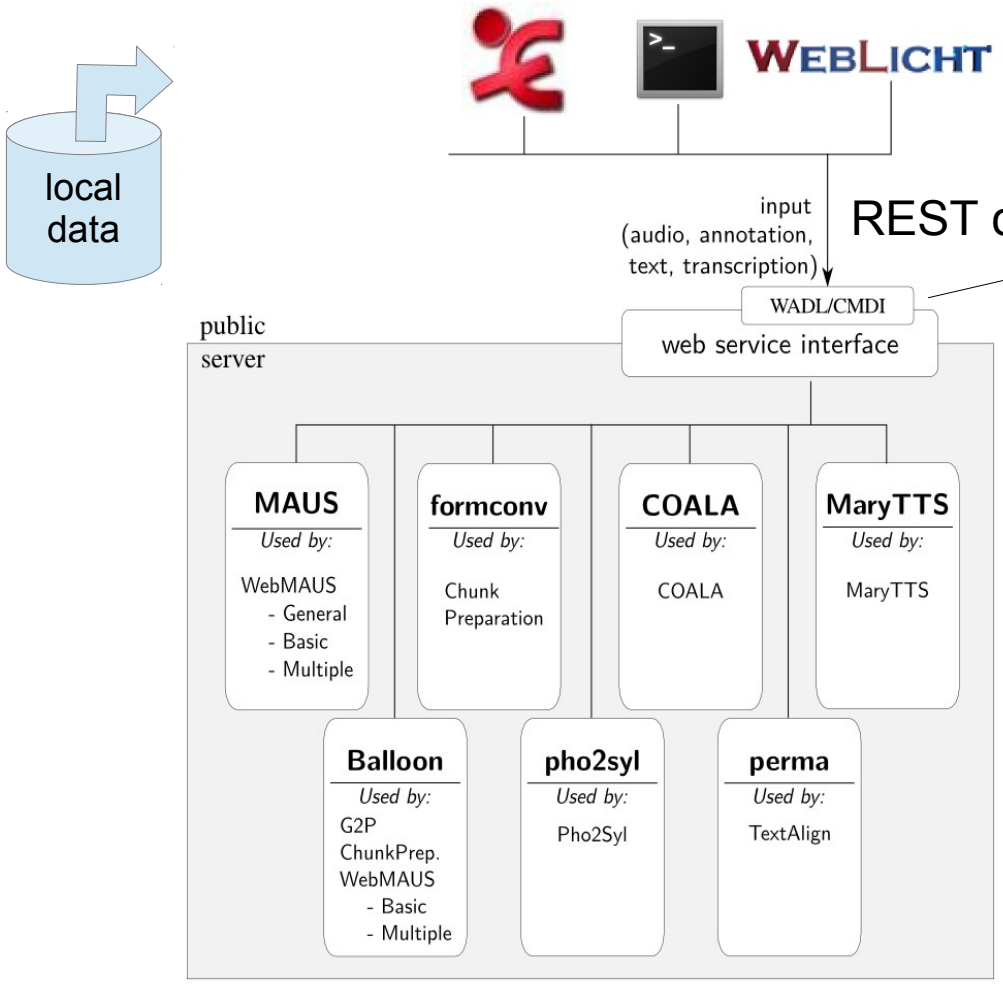
Basic concepts, definitions

- 'chainer' or 'orchestration':
  server connects web services in different locations (usually via REST calls),
  e.g. *Weblicht*

- 'local installation' ('standalone tool'):
  the 'offline' installation of a web service

- 'webservice metadata':
  describes what the services does, the input/output formats, the parameters of the service

BAS Webservices Overview




```
...
<Operations>
 <Operation>
  <Name>runMAUSBasic</Name>
  <Description>segments an audio file
into SAM-PA phonetic segments...
  </Description>
 <Input>
  <Parameter>
   <Name>SIGNAL</Name>
   <Description>mono sound file...
   </Description>
   <MIMEType>
    audio/x-wav</MIMEType>
   <isConfigurationParameter>
    false
   </isConfigurationParameter>
   <DataCategory>
 http://www.isocat.org/datcat/DC-2653
   </DataCategory>
   <displayName>Signal file
   </displayName>
...
```

clarin.phonetik.uni-muenchen.de/BASWebServices/

Implementation Backend Server

CLARIN-D

Backend server:

- implemented in JAVA + plus calls to helper programs
  + good development support (tomcat,eclipse), stable language
  - unsure future of JAVA, helper programs need to be maintained (problems when language is no longer supported)
- hardware: double redundant Linux server with 24 cores, 24GByte RAM, fast Internet (2 * 1GB)
  + stable (no outage in 6 years)
  + easy to port (already the second generation server)
  - requires professional maintenance (security!)
- logistics: parallel production and development server (2)
  + rollout of new versions after extensive testing
  - no fool-proof test routines possible
- metadata must always match implementation
  + automated generation of webinterface, help pages from metadata

REST call to Backend Server

CLARIN-D

Example for a REST call to a BAS webservice 'runMAUS':

```
curl -v -X POST
   -H 'content-type: multipart/form-data'
   -F OUTFORMAT=TextGrid
   -F NOINITIALFINALSILENCE=false
   -F INSKANTEXTGRID=true
   -F LANGUAGE=deu-DE
   -F TEXT=@<filename>
   -F INSORTTEXTGRID=true
   -F USETRN=force
   -F SIGNAL=@<filename>
   'https://clarin.phonetik.uni-
    muenchen.de/BASWebServices/services/
    runMAUSBasic'
```

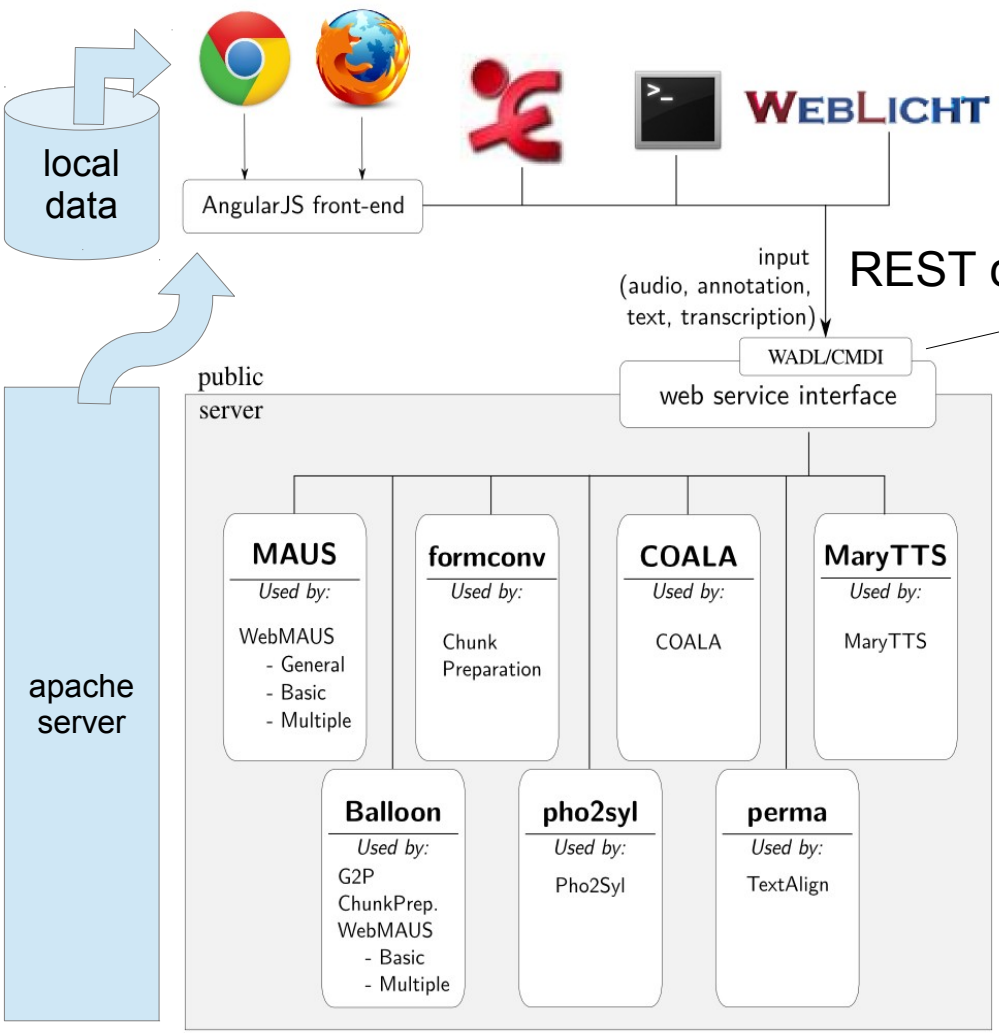
clarin.phonetik.uni-muenchen.de/BASWebServices/services/help

Implementation Webinterface

Webinterface:

- implemented in
  AngularJS + TypeScript (JS) + HTML5 + Plugins (JS)
  + good development support (eclipse), compatible with most browsers
  - future of AngularJS and TypeScript?
- about 90% of webinterface generated from webservice metadata (CMDI) at deployment
  + easy extensions, e.g. new parameters / parameter values
  - not always possible, e.g. when functions are not implementable as webservices, 'hacks' hard to maintain (e.g. batch processing webinterface → there are no batch-processing webservices → upload/download and batch loop must be implemented in the webinterface)
  - additional 'service elements', such as hoover messages etc. require a lot of implementation effort.

BAS Webinterface Overview




```
...
<Operations>
 <Operation>
  <Name>runMAUSBasic</Name>
  <Description>segments an audio file
into SAM-PA phonetic segments...
  </Description>
 <Input>
  <Parameter>
   <Name>SIGNAL</Name>
   <Description>mono sound file...
   </Description>
   <MIMEType>
    audio/x-wav</MIMEType>
   <isConfigurationParameter>
    false
   </isConfigurationParameter>
   <DataCategory>
 http://www.isocat.org/datcat/DC-2653
   </DataCategory>
   <displayName>Signal file
   </displayName>
...
```

clarin.phonetik.uni-muenchen.de/BASWebServices/

Advantages of webservices

CLARIN-D

No need for software distribution

- easier license management
- no support for different OS / versions
- much less user support
- better control of user access (e.g. via AAI)
- monitoring / usage statistics

Webservices can be embedded in

- user applications
- other CLARIN tools (e.g. Weblicht, ELAN)
- batch processes

Advantages of webservices: chaining

Example for a Weblicht chain:

- Weblicht uploads input data, here: txt and wav file
- Weblicht calls runMAUSBasic in Munich to calculate segmentation (TextGrid file)
- Weblicht calls converter service in Hamburg to convert TextGrid to EAF
- Weblicht calls visualization service at MPI Nijmegen to display result EAF

weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/Main_Page

Disadvantages of webservices

Work only when Internet available

- can be a legal problem when users are not allowed to upload protected data
- can be a problem in field work
  → some of the BAS webservices are available as local installations:
  SpeechRecorder
  MAUS
  EmuLabeller

References

References

Reichel, U.D. (2012). PermA and Balloon: Tools for string alignment and text processing, Proc. Interspeech. Portland, Oregon, paper no. 346.

Schiel F. (1999): Automatic Phonetic Transcription of Non-Prompted Speech, Proc. of the ICPhS 1999. San Francisco, August 1999. pp. 607-610.

Kisler, T. and Schiel, F. and Sloetjes, H. (2012): Signal processing via web services: the use case WebMAUS, Proceedings Digital Humanities 2012, Hamburg, Germany, Hamburg, pp. 30-34.

Schroeder, M. and Charfuelan, M. and Pammi, S. and Steiner, I. (2011) Open source voice creation toolkit for the MARY TTS Platform, 12th Annual Conference of the International Speech Communication Association - Interspeech 2011, Aug 2011, Florence, Italy. ISCA, pp.3253-3256.

Thomas Kisler, Uwe D. Reichel, Florian Schiel, Christoph Draxler, Bernhard Jackl, Nina Pörner (2016): BAS Speech ScienceWeb Services – an Update on Current Developments. Proc. of the LREC 2016, Slovenia.