Tool Discovery - Helping scholars find the tools they need

Maarten van Gompel, KNAW HuC

Introduction

How do we help scholars find the tools they need?

Challenges

Scholars face various challenges:

1. How to learn what tools CLARIAH has to offer? – How to get an *up to date* and *complete* overview of all tools produced in CLARIAH?
   - Projects like CLARIAH and predecessors (CLARIN-NL) produce a large amount of software tools
   - Which tools are even considered CLARIAH tools? How much legacy from the past do we want to carry along?
   - Existing portals are often *incomplete* and *out of date* (e.g. CLAPOP), too reliant on manual curation
   - Existing portals cover only a single institute or a subset of CLARIAH tools

How **NOT** to get a sensible overview of all tools produced in CLARIAH?

Manually compiled lists, shared ad-hoc, are not sustainable:

CLARIAH Tools and Data

File Edit View Insert Format Data Tools Extensions Help Last edit was seconds ago

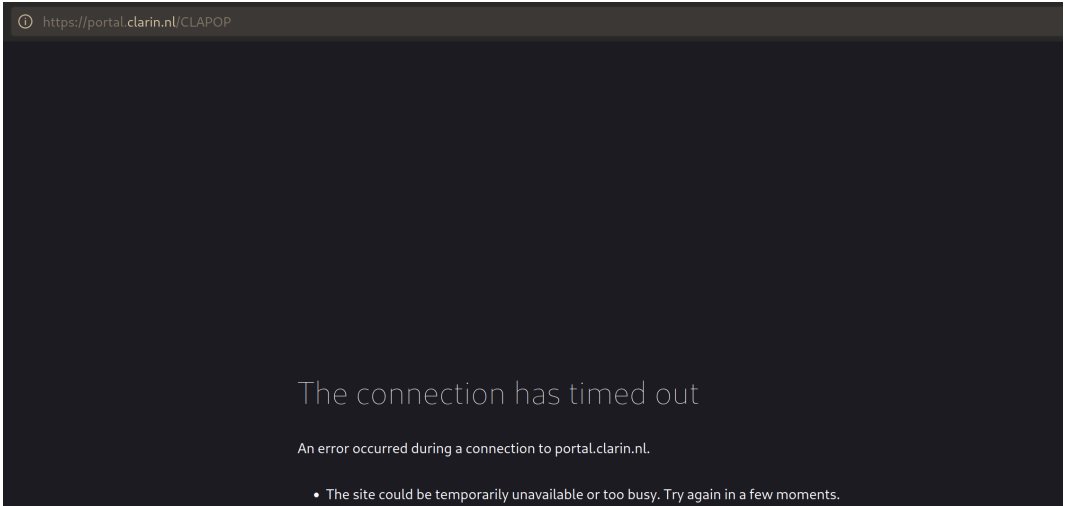
3:3 Alpino

	A	B	C	D	E
1	Name+link	Functionality	Type	WPs	Disciplines
3	Alpino	parsing, dependency parsing, lemmatisation, morphological analys	web application, local	WP3	Linguistics
4	Alpino (CLST web service and application)	parsing, dependency parsing, lemmatisation, morphological analys	web application	WP3	Linguistics
5	Anansi	central CLARIAH data connection		WP2	General
6	Anansi Query Interface	search for (anansi hosted) linked open data	web application	WP2	General
7	AntiLoPe Pipeline	NLP pipelines		WP3	
8	Automatic Transcription of (Oral History) Interviews / KALDI S	speech recognition, diarization	web application	WP3	Oral History, Political St
9	AutoSearch	corpus browsing, corpus searching, corpus processing	web application	WP3	Linguistics
10	Biblia Hebraica Stuttgartensia Amstelodamensis	Text and linguistic annotations of the Hebrew Bible in Text-Fabric fo	corpus data	--	Linguistics, Theology
11	BlackLab	Token-based corpus search engine	software library		
12	Bridge the Gap Arabic Corpora	search in Arabic Corpora	web application	WP3	Linguistics, History
13	CATTLE	conversion csv->rdf		WP4	Social Economic Histor
14	CHAMD	Conversion (CHAT->PEP-text), used in GrETEL 4		WP3	Linguistics
15	CHN	corpus browsing, corpus searching, corpus processing	web application	WP3	Linguistics
16	CLAAS	central CLARIAH Architecture		WP2	General
17	CLAM	Webservice framework	software library	WP3	
18	CMD2RDF	searching, browsing	web application	WP3	General
19	Colibri Core	n-gram & skip-gram tool	web application	WP3	
20	Corpus Gysseling	corpus browsing, corpus searching	web application	WP3	Linguistics
21	COW	conversion csv->rdf		WP4	Social Economic Histor

Existing portals are often *incomplete* and *out of date*

Why?: they rely on manual curation by a content maintainer

. . . or they are simply down altogether:



Existing portals are often *incomplete* and *out of date*

Why?: they harvest old information from other portals. Unnecessary middle-men

Virtual Language Observatory Search Contributors Help CLARIN

VLO / Faceted search / Search results / Record: Frog, an advanced Natural Language Processing Suite for Dutch

Frog, an advanced Natural Language Processing Suite for Dutch

Record details | Links (2) | Availability | All metadata | Technical Details

Name	**Frog**, an advanced Natural Language Processing Suite for Dutch
Description	**Frog** is an integration of memory-based natural language processing (NLP) modules developed for Dutch. All NLP modules are based on Timbl, the Tilburg memory-based learning software package.
Collection	INT
Modality	written
Country	Netherlands
National project	CLARIAH
Resource type	software
Data provider	Instituut voor de Nederlandse Taal

HDL 10032/198143d2010e74ae...

Landing page

Linked resource

frog

Existing portals only cover a single institute or a subset of tools

Challenges (2)

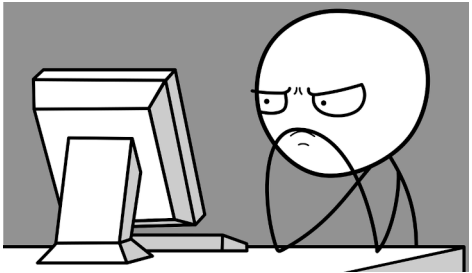
Scholars face various challenges:

1. How to get an **up-to-date** and **complete** overview of all tools produced in CLARIAH?
2. How to **identify** which tools are suitable for their needs?
   - Relies on availability of *accurate* and *complete* metadata
   - Software offers various interfaces, suited for specific audiences *(e.g CLI, Web application, web service, python module)*
   - Software may be too experimental
   - Software may be unmaintained/outdated

How to identify which tools are suitable for their needs?

Scholars will get frustrated when:

- Software doesn't install
- Software is buggy
- Software doesn't offer an appropriate interface
- Software doesn't make clear what problems it solves
- He/she has no idea how to use the software (lack of documentation?)
- There's nobody who can answer support questions, fix bugs (software not maintained?)



Out Mission

Our mission: We want to provide accurate **software metadata** so the user doesn't fall prey to these frustrations

Our solution (1)

1. How to get an **up-to-date** and **complete** overview of all tools produced in CLARIAH?
   - Developers know best how to describe their software *alongside their source code*; full agency; no man-in-the-middle
   - Periodic and automatic harvesting of software metadata *from the source*
   - Accommodate *existing* software metadata practises, map them to a *uniform vocabulary*.
   - Strong requirements to CLARIAH participants to include all their software to guarantee *completeness*

Our solution (2)

2. How to **identify** which tools are suitable for a scholar's needs?
   - To make this decision, metadata must be accurately reflect various aspects of the software, including:
     - Name, description, authors, maintainers & contributors
     - Support channels *(e.g. bug/issue tracker, mailing list)*
     - Licensing and access
     - The interface types *(command line? library? web app? mobile app?)*
     - Target platform *(Linux, Windows, macOS, web, mobile, python..)*
     - The development status *(actively maintained? abandoned?)*
     - Technology readiness level *(proof of concept? experimental? proven?)*
     - Links to documentation, release notes, screenshots
     - Associate publications
   - We provide a uniform way of describing this metadata as Linked Open Data (and require this from developers)
   - User must be given the ability to search on arbitrary metadata (e.g. faceted search)
   - Software must comply to certain software requirements ensuring a certain quality. We can automatically test compliance (to a limited degree) and communicate to the user whether these are met.

Deliverables

We deliver the following:

1. A **metadata harvesting pipeline**; software for harvesting and conversion from heterogeneous software metadata sources
   - **Tool Source Repository**; input for the harvesting pipeline, aimed at developers
2. A **Tool Store** that makes available (and searchable) all harvested metadata
   - Web interface for end-users (limited)
3. **Software Metadata Requirements**; document requirements and specifies the necessary *vocabulary*, aimed at developers

Tool Store: Example (1)

https://tools.dev.clariah.nl

Clariah Tools

Alpino

Computational Linguistics, University of Groningen

Alpino parser and related tools for Dutch [view more]

Source code

Created: 2019-10-09 Modified: 2022-06-15

Web API | Web Application | repo status Active

Alpino-Webservice 2.3

Maarten van Gompel

Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. This is the webservice for it. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document. [view more]

Internet > WWW/HTTP > WSGI > Application | Text Processing > Linguistic

dependency parsing | folia | linguistics | nlp | syntax

Source code

Go to Alpino *(WebApplication)*
https://webservices.cls.ru.nl/alpino

Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document.

Web API: Alpino
https://webservices.cls.ru.nl/alpino

Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document.

Created: 2015-09-08 Modified: 2022-04-08

repo status Active

analiticcl v0.4.1

Maarten van Gompel

an approximate string matching or fuzzy-matching system for spelling correction, normalisation or post-OCR correction [view more]

Tool Store: Example (2)

Clariah Tools

Cards | Table | SPARQL

Name	Version	Interface type	Description	Links	Status	Maintainer	Authors	Producer/Provider
Alpino	unknown 2022-06-15 11:59:11 +0200	Unknown	*Alpino parser and related tools for Dutch* [view more]	https://github.com/rug-compling/alpino	unknown			Computational Linguistics, University of Groningen
Alpino-Webservice	2.3 2021-01-08 22:21:09 +0200	Web API Web Application	*Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. This is the webservice for it. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document.* [view more] **Category:** Internet > WWW/HTTP > WSGI > Application; Text Processing > Linguistic **Keywords:** dependency parsing, folia, linguistics, nlp, syntax	https://github.com/proycon/alpino_clam_webservice	repo status Active	Maarten van Gompel	Maarten van Gompel Maarten van Gompel	
Alpino		Web API	*Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document.*	https://webservices.cls.ru.nl/alpino			Gertjan van Noord	
Alpino		Web Application	*Alpino is a dependency parser for Dutch, developed in the context of the PIONIER Project Algorithms for Linguistic Processing, developed by Gertjan van Noord at the University of Groningen. You can upload either tokenised or untokenised files (which will be automatically tokenised for you using ucto), the output will consist of a zip file containing XML files, one for each sentence in the input document.*	**https://webservices.cls.ru.nl/alpino**			Gertjan van Noord	
analiticcl	v0.4.1 2022-06-16 16:22:02 +0200	Unknown	*an approximate string matching or fuzzy-matching system for spelling correction, normalisation or post-OCR correction* [view more]	https://github.com/proycon/analiticcl	repo status Active	Maarten van Gompel	Maarten van Gompel	
Automatic Speech Recognition for Dutch	0.5.3	Web API Web Application	*This is a web-based automatic speech recogniser for Dutch, capable of transcribing dutch speech recordings using multiple models.* [view more] **Keywords:** dutch, nlp, speech recognition	https://github.com/opensource-spraakherkenning-nl/asr_nl	repo status Active	Maarten van Gompel	Emre Yilmaz Louis ten Bosch Maarten van Gompel	Centre for Language and Speech Technology , Centre for Language Studies , Radboud University
Automatic Transcription of Dutch Speech Recordings	0.5.0	Web Application	*This webservice uses automatic speech recognition to provide the transcriptions of recordings spoken in Dutch. You can upload and process only one file per project. For bulk processing and other questions, please contact Henk van den Heuvel at h.vandenheuvel@let.ru.nl.*	**https://webservices.cls.ru.nl/asr_nl**			Maarten van Gompel Emre Yilmaz	
Automatic Transcription of Dutch Speech Recordings	0.5.0	Web API	*This webservice uses automatic speech recognition to provide the transcriptions of recordings spoken in Dutch. You can upload and process only one file per project. For bulk processing and other questions, please contact Henk van den Heuvel at h.vandenheuvel@let.ru.nl.*	https://webservices.cls.ru.nl/asr_nl			Maarten van Gompel Emre Yilmaz	
BlackLab Corpus Search	2.3.0 2021-06-12 19:18:02 +0200	Unknown	*The parent project for BlackLab Core and BlackLab Server.* [view more] **Keywords:** corpus	https://github.com/INL/BlackLab http://inl.github.io/BlackLab/	unknown		Koen Mertens Jan Niestadt	Dutch Language Institute Instituut voor Nederlandse Taal (INT)
CLAM	3.1.5	Unknown	*Quickly turn command-line applications into RESTful webservices with a web-application front-end. You provide a specification of your command line application, its input, output and parameters, and CLAM wraps around your application to form a fully fledged RESTful webservice.* [view more] **Keywords:** natural language processing, nlp, rest, webservice	https://github.com/proycon/clam https://proycon.github.io/clam	repo status Active		Maarten van Gompel	Centre for Language and Speech Technology , Centre for Language Studies , Radboud University Humanities Cluster , KNAW

Figure 7: Tool Store - Table

Tool Store: Example (3)

Clariah Tools

Cards Table SPARQL

fusus

Workflow for converting Arabic scanned pages into readable text

Go to the software website
https://github.com/among/fusus

Properties

Version
0.0.2 (release notes)

Interface types
Unknown

Software website
https://github.com/among/fusus
https://among.github.io/fusus/fusus/index.html

Source code repository
https://github.com/among/fusus stars 3

Category
Religion | Scientific/Engineering > Information Analysis | Sociology > History | Text Processing | Text Processing > Fonts | Text Processing > Markup

Keywords
arabic | digital-humanities | image processing | image-processing | islam | kraken | medieval | OCR | ocr | opencv | python | text | text-fabric | text-processing | wisdom | workflow

Development Status
repo status WIP

Issue Tracker (Support)
https://github.com/among/fusus/issues issues 0 open issues 3 closed

License
MIT

Author(s)
Cornelis van Lit
Dirk Roorda
Cornelis van Lit, Dirk Roorda

Producer
Among, A Community for DH and MS

Runtime Platform
Python 3
Python 3 Only
Python Implementation CPython

Operating System
MacOS > MacOS X
Microsoft > Windows > Windows 10
POSIX > Linux

Objectives

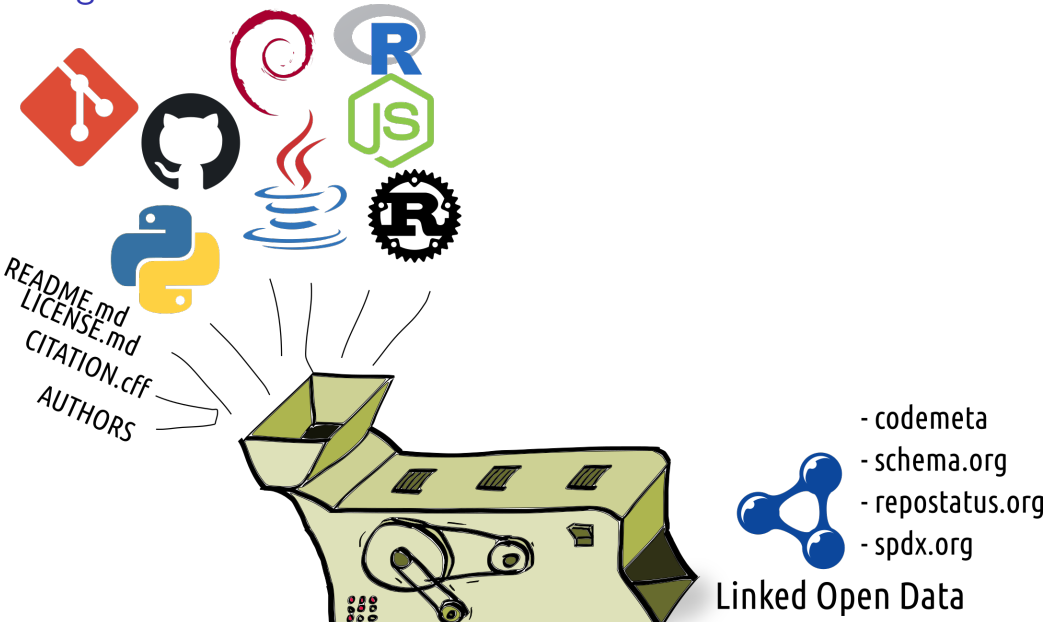
Our objectives from a more technical perspective:

1. Ensure software providers need to specify their metadata *only once* and can reuse *their* existing sources as much as possible
   - Ensure software providers need to register their software for inclusion in CLARIAH only once (Tool Source Repository)
2. Establish a single well-documented unified vocabulary for our software metadata needs; automatically convert to that
   - Fully embrace Linked Open Data as a standard
   - Build upon existing initiatives
3. Short automatic update cycles (harvesting at regular intervals)
4. Provide an API and/or export abilities for integration with other tools (e.g. Ineo)

Technologies

- **Linked Open Data**; all harvested software metadata is made available as Linked Open Data
  - we use JSON-LD and Turtle for serialisation, we provide SPARQL endpoints for querying
- **schema.org** and **codemeta**; we build upon these main vocabularies
  - Conversion from heterogeneous existing metadata formats for software: `setup.py` (Python), `pyproject.toml` (Python), `package.json` (js), `pom.xml` (Java/Maven), `Cargo.toml` (rust)
- **repostatus.org**, **spdx.org**; additional vocabularies we use for certain terms
- Where necessary we propose new vocabularies in collaboration with the wider community

Technologies: Harvester and Converter



Technologies: Full pipeline

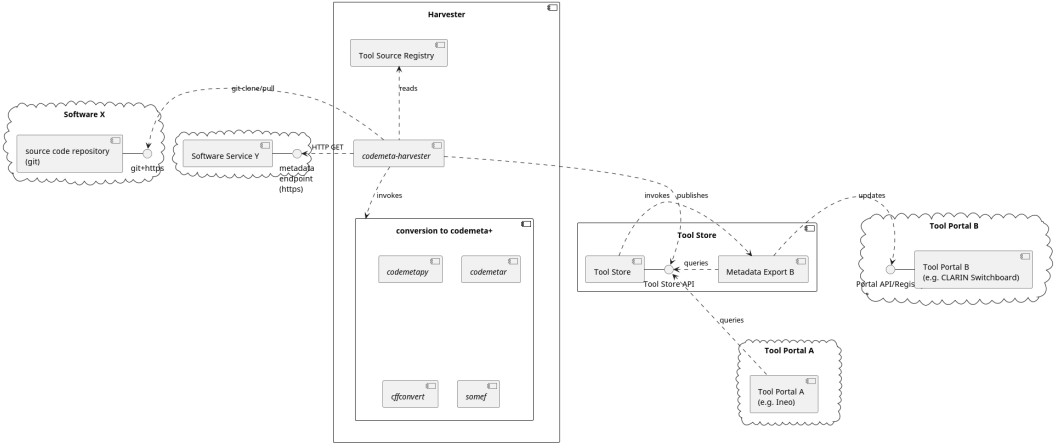


Figure 10: Tool Discovery Component Diagram

Collaboration

- Relation with **Codemeta**, **schema.org**:
  - We use their vocabularies and build extensions where needed, which we contribute back
  - Seeking embedding within the wider community
- Relation with **Ineo**
  - We provide the data feeding Ineo *automatically* and *regularly*
  - Ineo will act as front-end for a (subset) of our data
- Relation with the *Research Software Directory* (eScience):
  - Initial talks on establishing common representations and making our tools interoperable
  - The RSD might provide a user-friendly way for authoring software metadata manually but with smart automatic assistance
- Relation with CLARIN:
  - Output from earlier projects is considered in vocabulary-choices
  - Export towards CLARIN's infrastructure (CMDI and CLARIN switchboard) is on the agenda