

CLAM

From NLP command-line tools to webservice: current state of affairs



LANGUAGE MACHINES

CLST | Centre for Language and
Speech Technology
Radboud University



Introduction

Observation: NLP tools are often command-line programs ... for good reason.



Command line tools: pros

Command-line tools are a good thing!

"This is the Unix philosophy: Write programs that do one thing and do it well. Write programs to work together." (Doug McIlroy)

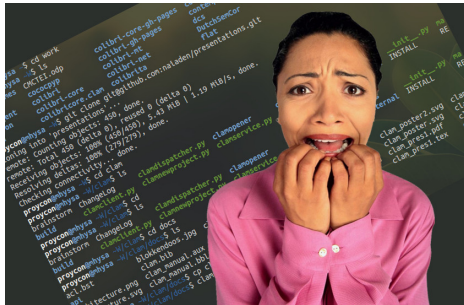
- **Flexibility & Extensibility:** Integrate tools into pipelines, the output of one tool is the input to another
- **Performance & Simplicity:** Little overhead
- **Modularity:** Separate the interface (GUI, web) from the actual program



Command line tools: cons

But...

- The command-line is challenging and intimidating for non-technical end-users



- **Installation:** Installation may complex and depend on other software
- **Connectivity:** Web-connectivity/networking has to be explicitly built-in into your program (not trivial)

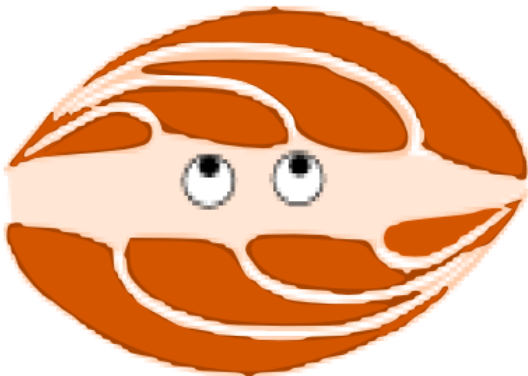
Web-connectivity through CLAM

Objectives

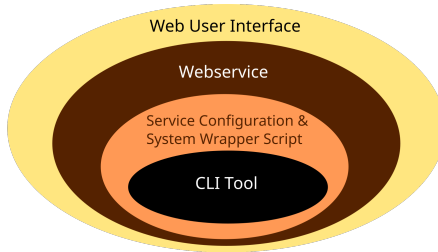
- Quickly transform existing command-line tools into fully-fledged webservices
 - No need to alter the tool itself
 - ▶ No requirements on programming language or technology, as long as it runs on Linux/BSD and can be invoked from the command line
 - Maintain flexibility and modularity
 - Simply describe the tool in terms of input, output and parameters using CLAM
- Offer a generic **RESTFUL interface** for machines
- Offer a generic **Web-based User Interface** for human end-users
- Deal with batch processing and large data: NLP tasks may typically run for a long time on large corpora



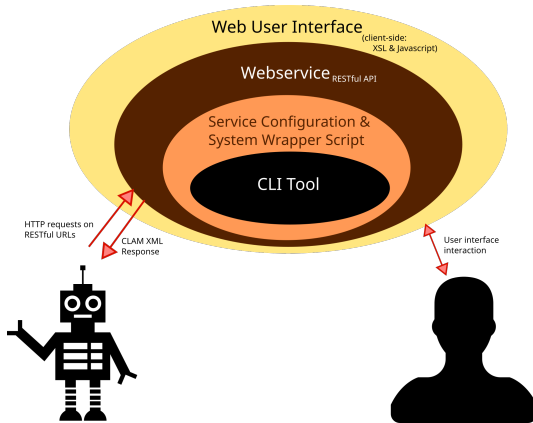
Architecture



Architecture



Architecture



Providing a Service (1/2)

In order to wrap a tool and make a webservice:

1. Write a service configuration file

- General meta information about your system (name, description, etc. . .)
- Definition of global parameters accepted by your system
- The program to invoke (i.e. the wrapper script around your NLP tool)
- Definition of *profiles*
 - A profile defines in detail what output files a system produces given certain input files.



Providing a Service (2/2)

In order to wrap a tool and make a webservice:

2. Write a wrapper script for your system

- Wrapper script is invoked by CLAM, and should in turn invoke the actual system
- Acts as glue between CLAM and your NLP Application.
- Can be written in any language (python users may benefit from the CLAM API)
- Not always necessary, simpler command-line applications can be invoked directly by CLAM as well.

Development

CLAM has a built-in webserver so it can be tested quickly in development.



Profiles

- **What output files are produced given which input files?**
- What format are the input files in? (CLAM needs not be able to parse it itself)
- What parameters (metadata) are required or possible on input files?
- How is metadata propagated from input files to output files?
- What viewers are associated with output files?
- Which converters can act upon input/output files?



Resources

Resources

- **Projects:** Stored on server, owned by a user, each corresponds to a single run of the tool
 - **Global parameters:** Parameters for the run
 - **Input files:** Upload files or choose from preset collections on the server
 - ▶ Local parameters (i.e. metadata)
 - **Output files:** Can be downloaded as-is, visualised using a *viewer* or external webservice.



Workflow

Typical Workflow

1. Authentication
2. Create a new project
3. Upload files (and set per-file parameters if applicable)
4. Set global parameters
5. Start the project
6. Wait for completion
7. Download/view output files
8. Delete project



Text Statistics (CLAM Demo)

test

Status

Accepting new input files and selection of parameters

Abort and delete project

Input

Input files

Show 10 entries

Search:

Input File	Template	Format	Actions
test.txt	Input text document		

Showing 1 to 1 of 1 entries

First Previous 1 Next Last

Upload a file from disk

Use this to upload files from your computer to the system.

Step 1 First select what type of file you want to add:

Step 2 Set the parameters for this type of file:
Select a type first

Step 3

Grab a file from the web

Retrieves an input file from another location on the web.

Step 1 First select the desired input type:

Step 2 Set the parameters for this type of file:
Select a type first

Step 3 Enter the URL where to retrieve the file

Step 4

Add input from browser

You can create and add new files from within your browser:

Parameter Selection

Main

Create Lexicon

Generate a separate overall lexicon?

☐

Case Sensitivity

Enable case sensitive behaviour?

Limit frequencylist

Limit entries in frequencylist to the top scoring ones. Value of zero (no limit) or higher

Author

Sign output metadata with the specified author name

Powered by CLAM - Computational Linguistics Application Mediator
by Maarten van Gorpel
Induction of Linguistic Knowledge Research Group, Tilburg University



Projects vs Actions

Projects: Batch processing

- CLAM is optimised for **batch processing**, your tool may run for hours or days if necessary
- The user can always close his browser and come back later
- Data stored and held on server until explicitly deleted
- Different from real-time request-response cycles

Actions: Real-time response

- Define a command line application or Python function to run for a specific webservice URL
- Independent of projects
- Extensive parameter specification (but no file upload!)
- Command/function is expected to return a result in a short time interval
- Output of command/function is returned by CLAM to the user/client



Authentication

Authentication

- Projects are user-specific
- Authentication support through:
 1. HTTP Digest Authentication
 - ▶ Explicit user specification in configuration
 - ▶ or database-backed (MySQL)
 2. Pre-authentication by webserver (usable with for instance Shibboleth)
 3. OAuth2



Future work

Future work

- Port of underlying framework from web.py to Flask
- Python 3 support
- Testing in CLARIN authentication infrastructure



Demo

- CLAM website: <http://proycon.github.io/clam>
- Numerous webservices from our department are hosted here:
<http://webservices-lst.science.ru.nl>
- (register for a free account if you have none yet)

