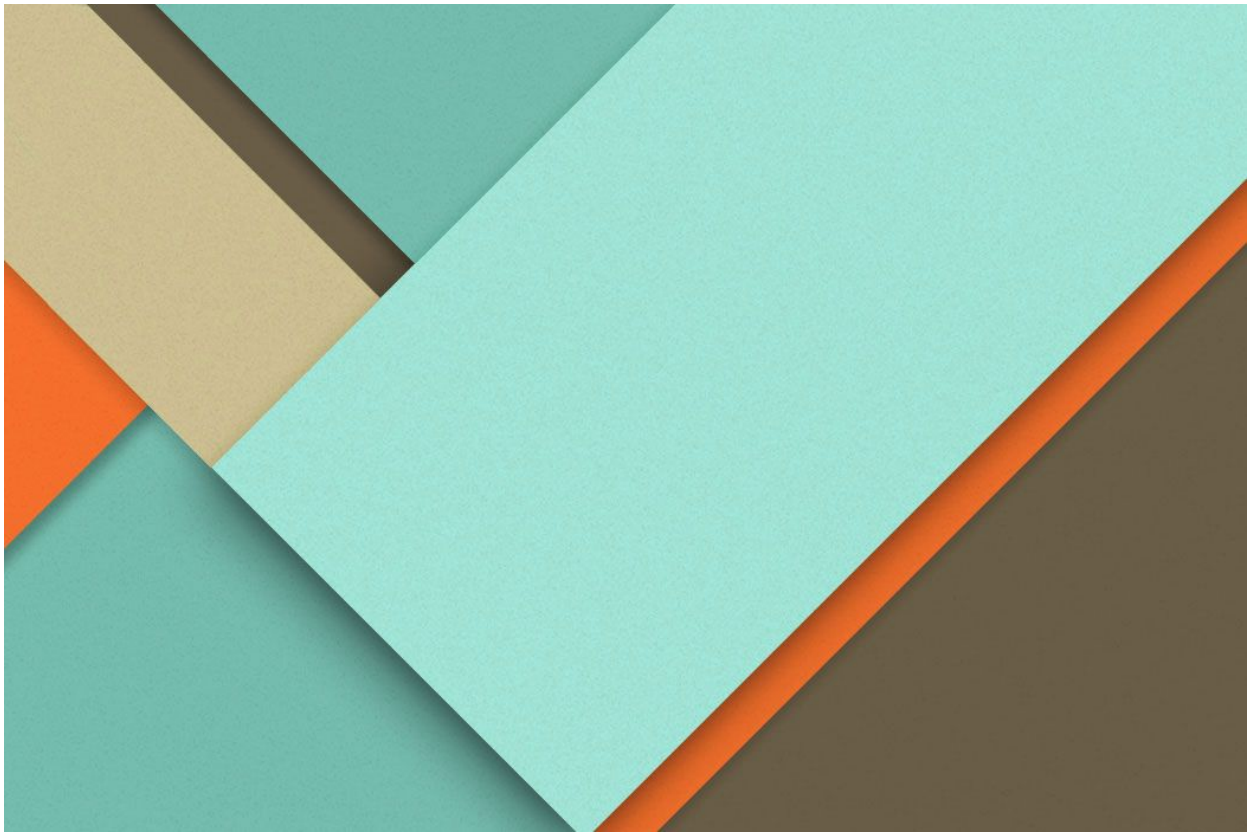

TEI export for the CDLI corpus

2019 April 9

Project application to the Cuneiform Digital Library Initiative Google Summer of Code.

Ralph Giles

+1 (604) 352-5431

giles@thaumas.net

rillian

https://gitlab.com/rillianbis/

304-1234 Pendrell St

Vancouver BC V6E 1L6

Canada

Synopsis

Recently, the Perseus Digital Library project published their next-generation online interface for reading and comparing texts, called Scaife. While Perseus is focussed on ancient Greek and Latin, the viewer project is separate and intended to be available for other languages.

I propose to write an export tool for the CDLI dataset so it can be used with this new viewer. The tool will need to convert the native AFT markup used by CDLI to the Text Encoding Initiative (TEI) XML schema used by Perseus, and generate corresponding Canonical Text Services annotations for each source.

This will improve the accessibility of the CDLI corpus by making it available to the larger community of tools developed for TEI analysis and provide a newer, more powerful, option for accessing the CDLI corpus online.

About the project

The primary objective of this project is to produce a tool for converting AFT to TEI. Focussing on this enables future work in a number of different directions, regardless of the success of other aspects of the project.

- **Command-line tool** for converting ATF to TEI
    - Start with the ply-based parser for ATF in pyoracc.
        - Builds on existing work.
        - Python is widely used by NLP folks, so more accessible and maintainable code.
        - I noticed in passing pyoracc doesn't validate the current CDLI database dump, so some cleanup work is likely here.
    - If Python is too slow, I prefer porting the tool to Rust.
        - Pro: At least as fast as C/C++, high-level.
        - Con: New language, less accessible to casual developers.
        - Implement new ATF parser library on top of nom.
        - Port rest of conversion logic.
- Investigate **schema mappings**.
    - Oracc has also done some work here, which is a good starting point.

    - Have tablet metadata, transliteration, translation, images, museum and publication references.
    - Any other requirements?
- Stand up an **instance** of the Scaife Viewer.
    - Build familiarity with features and tools the converter should support.
    - Lets us iterate quickly and simplifies user testing and review.
    - Add the sciafe containers to the cdli deployment?
- **Document** tools and schema mapping.
- **Package** results to support reuse.
- Add **image support** to the Scaife viewer
    - Scaife is text-only at the moment.
    - It's great to compare with images and drawings of primary sources.
    - CDLI has lots of those; see if we can hook them up as another layer.
    - Supports better integration of the visual ml results.
- Investigate inclusion of **CoNLL** annotation layers.
    - Does it make sense to define a TEI div class for these?
    - Can scaife do anything useful with them? Could it?
    - Would adding direct support to scaife for CoNLL or RDF be better?
- Investigate implementing the **CTS api** as a web service.
    - Hyperlink directly to resources on the main CDLI site.
- Maybe produce an alternate CDLI export repo with TEI versions suitable for import.
    - Similar to the list Perseus uses.

I have a couple of other ideas which might work as side projects either as a warm-up or as time permits.

- Improve **search interface** on the existing site.
    - Search suggestions for fields like language and period.
        - Currently hard to know what to put there as a new users.
        - Query database for most common values.
        - Clarifies abbreviations, etc.
    - General keyword search.

        - Search in all fields, probably with a boost list for ordering results.
        - Can sort terms matching completion list above into associated fields.
- Use **sign list** on the CDLI website.
    - Allow searching transliterations by unicode cuneiform.
    - Allow optional display of transliterations as a sort of normalized transcription.
    - May not be popular with experts, but a helpful resource for outreach, students.
    - Assumes there's a digital sign list available. Can be best effort for now.

About the student

I'm studying the languages and cultures of the ancient Mediterranean at the University of British Columbia. I have had one term of Akkadian (Huehnergard's grammar through chapter 16) and a year of ancient Greek as of the beginning of the Summer of Code project. I've also read some Middle Egyptian on my own.

I'm primarily interested in ancient writing systems and making scholarship more accessible. Motivated by the promise of automation to make new kinds of research possible, I want to work with the CDLI as one of the most open ancient script collaborations and as a way to continue improving my knowledge of cuneiform.

Open Source experience

I have significant experience with open source. I'm used to developing code in collaboration with others and in public. I've worked as a paid developer on the Firefox web browser, primarily on audio and video playback support. I'm able to work independently and I have good collaboration and project management skills.

While at Mozilla, I spearheaded a project to ship the first code written in the new Rust programming language within Firefox, leading the way for multiple subsequent projects enabling improvements in performance and reliability. Introducing a new compiled language was a significant challenge is a project that size, with hundreds of active contributors and complex process automation.

Before that I worked on the Ghostscript project, a converter for Postscript and PDF document formats. In parallel, I volunteered with the Xiph.Org Foundation, helping develop and distribute the Ogg music format and the Opus audio compression technology now used by Youtube, Facetime, and Facebook Messenger, and part of the WebRTC standard for online communication.

Programming

I've written in Python, Rust, C, C++, Javascript, and unix shell. Most of my experience is with lower-level or backend code, but I have extensive experience administering servers and have developed basic websites in the past.

I hope you will consider my application and look forward to the possibility of working with you.