These are preliminary notes based on a first look at the needs, the previous assessments of tools and contemporary systems and the code that runs hypothes.is.

The hypothes.is system requires a login but, at present, names are being reserved. The add-on itself has the functionality to create a new account to use for annotation inside of the user interface.

Hypothes.is has a stream of tags - eg.

<https://hypothes.is/stream?q=tag:tests>

annotations can be flagged as public and private. This allowed linking through to another article wherein the annotations have common tags - <http://www.cplong.org/2014/06/seeding-publics-from-a-world-of-readers/>

When the annotation is publicly available, activating the add-on delivers that annotation irrespective of the author. The annotations are available to the open public regardless of whether they have logged in or not.

To add in the leaderboard concepts and qualifying the quality of comments, hypothes.is would have to be augmented. That can be done by forking the code and in a variant we would develop (ie. the one connected to an OJS implementation). Its code could be expanded to allow user profile hooks. The hooks could lead to an in-house source of metrics-- like that seen with the stats of Twitter users held inside of Twitter. The profile hooks could lead to a third party system that will evaluate metrics and report the quality of the comments (e.g., how klout.com will evalutate mulitple social media identities to establish a single assessment of an identity who participates in multiple networks).

There are multiple annotation systems in play (as per the Peer Review Personas piece describes). An active participant in one system may behave different in another system-- they may see to have little authority or few contributions if they are new to an open annotation system, but may have had in depth interactions in another system. There may never be a prevailing victor in the open annotation movement. If there will competition, favourites and new players, then they could use a standard for sharing persona activity to allow for a standardized set of inputs for reputation assessment.

One concept: a consistent set of qualifiers that shed light on a reviewer’s behaviour and reputation. Those could be offered by all open annotation systems (with the user’s permission to disseminate) with a consistent API response. Clearing house / reputation systems operating as third party sites could become known as a side effect of the authentication process.

e.g., if “aknow.com” requests statistics for <https://hypothes.is/stream?q=user:dewolfe001> then, hypothes.is would know that the third party site, aknow.com would hold an aggregate for the activities that hypothes.is knows as dewolfe001. Hypothes.is could next link to the third party profile on aknow.com. The API interaction could be a two-way street. statistics gathered for aggregation and the assessment of interactions, could be offered back programmatically as well.

Component Release Priorities and Thoughts

1. Marginal comments - hypothes.is is website agnostic, requiring an add-on to be installed in your browser (Firefox’s add on was assessed). It requires a sign-up / login. That could be replicated inside of a module for OJS as per the “OJS Project - Meeting Notes and Next Steps” email from July 31st. That plugin should be largely delivered on the client side using Javascript functionality. The in-broswer annotations would be similiar to how it is done with hypothes.is: in the client as a browser add-on). The email outlines an implementation that would be wholly contained along with our implementation of OJS. It would not be difficult to still lean on hypothes.is and its centralized system for coordinating users and their contributions. The setback of interoperating with the hypothes.is site and its services is the absence of apparent features to support review ratings and reputation management.
2. hypothes.is would need disambiguation added. ORCID looks really strong. Other systems (eg. OpenID) use multiple sources to identify and disambiguate personas.
3. Tagging is built into hypothes.is. Voting is not a part of hypothes.is and it does not have that element in its roadmap.
4. Article level review and rating are not part of the roadmap of hypothes.is. This article, <http://readwrite.com/2012/03/02/hypothesis> talks about reputation but it does not point to the specifics. I did ask via Twitter, but as of this writing, I have not see a response.
5. Text editing and revisioning can happen inside of a given CMS. That is outside of the scope of what hypothes.is offers. Version histories have to be declared in some fashion. If the URL changes, that’s clear way to consider two versions as distinct. If that happens, the previous reviews will become decoupled. The meta tag “revised” in HTML can be used to denote the revision. Web pages (i.e. journal articles in review) can be assessed for their URL coupled with their revised date. All comments would below to distinct edition.
6. An API to pull the actions of given reviewers through automation would allow stats to be gathered on participants who review articles and use an annotation system such as hypothes.is. The Persona site could hold these stats. It could return its data for use by other parties. It could also return constructed HTML for presentation by hypothes.is and similar. The return of consistently formatted data could mean that profile and community pages displayed by other annotation services could hold a consistency. Hypothes.is documentation (<https://hypothes.is/blog/cross-format-annotation/>) references its API, but the example given is in an available / active URL.

## Notes on Hypothes.is Licensing

I did notice the request to register with a licensing agreement:

<https://github.com/hypothesis/h/blob/master/CONTRIBUTING.rst#licensing>

If there is code contributed into the baseline / main project, the licensing has language to cover off a surrender of IP (patents, etc.) to the hypothes.is project. This may bring an additional benefit to the forked variant wherein the reputation management (votes on comments, etc.) is exclusive to our code.

## Notes on Elastisearch

Elastisearch is one of the required technologies used by Hypothes.is and Annotator.js. It is a Java driven web service that stores and produces schema-less data in JSON format.

For example, these two “curl” functions call an instance of Elastisearch running on my desktop.

curl -XPUT 'http://localhost:9200/twitter/user/Shawn' -d '{ "name" : "Shawn DeWolfe" }'

curl -XPUT 'http://localhost:9200/twitter/user/Nina' -d '{ "name" : "Nina Belojevic" }'

The PUT to the url (e.g. http://localhost:9200/twitter/user/Shawn deposits data (the “-d” arg precedes data). The string that is introduced (e.g. '{ "name" : "Shawn DeWolfe" }') is put into the data. It will overwrite the existing index’s value with this new data. By calling that same index and key (http://localhost:9200/**twitter/user/Shawn**), it will produce the stored data.

The elements of stored in a tree of files inside of the data directory related to the elasticsearch install. While MySQL and other SQL databases will store large all records of a table a set of two or three table-specific files, elasticsearch stores each record in its own set of files. The ability to write and read these files appears much faster that was I would expect to see in a system that had to traverse so much of the file allocation table (FAT).

## Notes on OJS

OJS is built from ground up using common libraries (eg. ADODB) and concepts (e.g. MVC).

## November 3rd, 2014 notes

<https://github.com/openannotation/annotator/blob/master/doc/getting-started.rst> -- the “Adding More Plugins” maps out how to add things like the Tags add-on, but that would also cover how to add something like the reputation management piece.

The hypothes.is API appears to run in our install, even it it doesn’t find any data:

<http://helene.library.uvic.ca:5000/api/search?limit=1000&text=this>

This material from the README is applicable:

===============

API Endpoints:

===============

\*\*/search\*\*

Search for annotations annotations

Examples:

https://api.hypothes.is/search?limit=1000&uri=http%3A%2F%2Fepubjs-reader.appspot.com%2F%2Fmoby-dick%2FOPS%2Fchapter\_003.xhtml&user=acct:gluejar@hypothes.is

https://api.hypothes.is/search?limit=1000&user=acct:gluejar@hypothes.is

https://api.hypothes.is/search?limit=1000&quote=limber

https://api.hypothes.is/search?limit=1000&text=consider

params:

\* limit - number of results to return

\* uri - url encoded uri to get annotations for

\* user - get annotations for a particular user. syntax: acct:<username>@<provider> . Until there are other annotation providers, the provider is "hypothes.is".

quote - words that the annotation is quoting. This is very brittle - text is not completely indexed, is punctuation sensitive and appears to index single words only. change your search word to lower case.

\* text - search annotation text. This is very brittle - text is not completely indexed, is punctuation sensitive and appears to index single words only. change your search word to lower case.

\*\*/annotations\*\*

https://api.hypothes.is/annotations/<annotation id>

method: GET

get an annotation

Examples:

https://api.hypothes.is/annotations/utalbWjUQZK5ifydnohjmA

method: POST

create a new annotation (needs authentication)

params:

refresh - a boolean that forces a refresh

method: PUT

update an existing annotation (needs authentication)

params:

\* refresh - a boolean that forces a refresh

method: PUT

delete an existing annotation (needs authentication)

\*\*/search\_raw\*\*

Advanced search API - direct access to ElasticSearch. Uses the same API as the ElasticSearch query endpoint.

.. toctree::

:maxdepth: 1

api/resources