2020-07-29 - Special Topic - VIVO Scholar Next Steps

Date

29 Jul 2020

- Time: 10:00 am, Eastern Time (New York, GMT-04:00)
- See in your timezone

Call-in Information

To join the online meeting:

- Go to: https://lyrasis.zoom.us/my/vivo1
- One tap mobile:
    - US: +16699006833,,9358074182# or +19292056099,,9358074182#
- Or Telephone:
    - US: +1 669 900 6833 or +1 929 205 6099 or 877 853 5257
    - Meeting ID: 935 807 4182
- International numbers available: https://zoom.us/u/aeANHanzED

Slack

- https://vivo-project.slack.com
    - Self-register at: http://bit.ly/vivo-slack

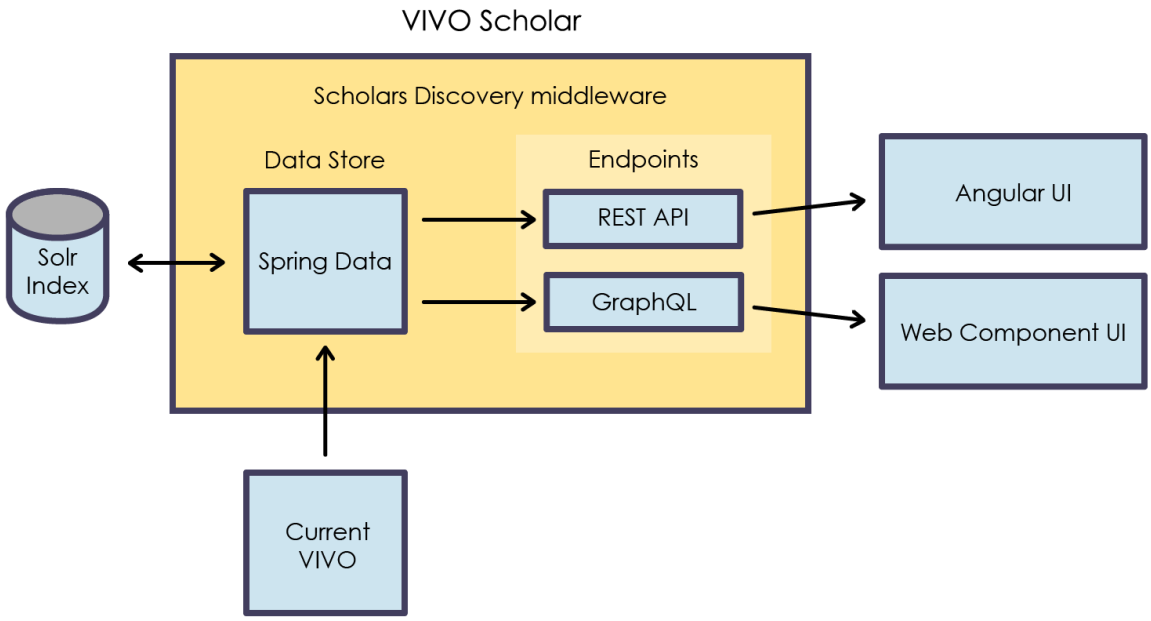
Attendees

⭐ Indicating note-taker

1. Andrew Woods
2. Brian Lowe
3. Ralph O'Flinn
4. Huda Khan
5. Benjamin Gross
6. William Welling
7. Alexander (Sacha) Jerabek⭐
8. ...

Objective

1. Work towards community decision on how to move the VIVO Scholar initiative forward



Agenda

1. Brief review of options to explore
    a. Single Solr for both VIVO and Scholar
    b. Option 1: populating Solr directly from VIVO
    c. Option 2: populating Solr via notifications framework
    d. Extra: potentially extracting indexing code into its own library
2. Review of Scholar's indexing approach
3. Review of VIVO's indexing approach
4. Process for technical exploration towards evaluating level of effort
5. Next steps

Notes

Draft notes in Google-Doc

Notes from chat:

From Benjamin Gross to Everyone: 10:14 AM

https://github.com/vivo-project/Vitro/blob/master/api/src/main/java/edu/cornell/mannlib/vitro/webapp/searchindex/documentBuilding/SelectQueryDocumentModifier.java#L90

Example modifiers holding queries: https://github.com/vivo-project/VIVO/blob/master/home/src/main/resources/rdf/display/everytime/searchIndexerConfigurationVivo.n3#L139

From Huda Khan to Everyone: 10:22 AM

this seems to be the main indexing change listener but I may be missing something: https://github.com/vivo-project/Vitro/blob/master/api/src/main/java/edu/cornell/mannlib/vitro/webapp/searchindex/IndexingChangeListener.java

I had experimented with tacking on linked data notifications but did that at the simple reasoned level (since that too is listening for changes to statements), but that may not be the ideal place.

https://github.com/vivo-project/Vitro/blob/master/api/src/main/java/edu/cornell/mannlib/vitro/webapp/searchindex/SearchIndexerImpl.java

From Benjamin Gross to Everyone: 10:39 AM

Getting the triples into the triplestore is comparatively fast so I don't think it would be particularly bad to always pause search indexing during a load...?

From Brian Lowe to Everyone: 10:41 AM

It should already be doing that, e.g. if you upload a large file it should pause the indexer until the writing is done. The problem is that for a lot of people a "bulk load" is a series of SPARQL UPDATES where the indexer is unpaused after each one.

From Huda Khan to Everyone: 10:41 AM

babies will baby, it's ok

From Benjamin Gross to Everyone: 10:43 AM

Hmm I see. Why does the RDF upload form seem to time out when uploading large files? The system is waiting for something before returning a response, presumably.

From Brian Lowe to Everyone: 10:46 AM

The response is returned when the reasoning finishes and the indexing starts (the reasoning is synchronous; the indexing is asynchronous). Timeouts are usually a problem between Apache and Tomcat; there's a default (something like 5 minutes) that mod proxy will wait for a response from Tomcat. Using Tomcat directly on 8080, you usually won't see timeouts.

From Benjamin Gross to Everyone: 10:48 AM

Ah, it's the reasoning then. Okay, thanks. It would be nice if the reasoning happened in the background and allowed the user to continue going about their business, but not a high priority.

From Brian Lowe to Everyone: 10:49 AM

For bulk uploads, yes. The way it currently works is designed for self editing, where if you see a page before the inferences are added it might look like things are missing.

Aw: trying to get some clear ideas on what to explore so that we have some indications on how to proceed when we meet again.

Possible options:

a. Single Solr for both VIVO and Scholar
b. Option 1: populating Solr directly from VIVO
c. Option 2: populating Solr via notifications framework
d. Extra: potentially extracting indexing code into its own library

Aw: do we want to hardwire solr, or have a more generalized messaging approach?

Rof: potential rewrite ahead if we go ‘everything messaging route’, mixing an matching is not advisable. May not have resources to do rewrite of messaging.

Aw: so hardwired approach is recommended? Not sure why rewrites are necessary?

Rof: should expand what we have now, and then adding messaging approach later.

Ww: messaging is more mid to long term goal, most expedient path is to use modifiers etc. to populate Scholar’s Discovery. But messaging triples might not be the most practical option. Would go for option of populating Solr directly from VIVO (Option 1 above).

De: would support most expedient approach, just get it done.

Aw: event driven flows focused on messaging might not be the best option here. So, Option 1 would be the best way to go. And then bring in a messaging structure afterwards

Bl : we were leaning in that direction originally.

Ww: also would support Option 1.

Bl: uri finding can trigger triple changes

Aw: not clear about indexing change vs document modifier

Hk: [explains some experiments done with various methods for updating triple changes using simple reasoner, does search index listener do the same thing? How to determine how changes are identified and processed.

Bl: would like to use what works with current infrastructure without having to reinvent the wheel. Should try to fit Scholar into existing architecture that we know works and use it for a new purpose.

Aw: sounds reasonable. Need to look more into some of the classes like query document modifier and see how it takes advantage of lower level infrastructure.

De: important to note that we will have to have new documentation as the structure gets more complicated for batch uploads etc.

Bl: it will add time to indexing, but at some point we need to create new java classes. Maybe a new search indexer. But simpler to do it at the level of select query and document finder (?). Lots of little queries get run to create a document, may be able to grab bigger chunks of data or json blobs to speed things up. Not necessarily blow up indexing time, there are options to keep time reasonable.

De: can suspend indexing during bulk uploads

Ww: makes sense to pause indexing during bulk uploads

Aw: what are we shooting for? What is VS expecting? We are trying to figure out how to reuse VIVO core to populate index.

Ww explains: may not have to recode that much, (shows VS managed schema, Solr config). Ideal to have a flattened document; whole strings are used, tokenized strings good for full text search, nested versions get tricky. Elastic search supports replacement patterns, so when indexing whole strings can handle fragments (?). Date recognizer transforms dates to allow for better Solr indexing.

Ww: spring annotated models, sort annotations, document is used to provision Solr. Not all data is typed, some types need to be inferred. 7 different classes, critical top level entities, are used to make up a document. (see GitHub /vivo-community/scholars-discovery…)

Ww: how things get flattened is the key piece to look at and understand if we want to re-use VIVO infrastructure.

Hk: sounds like you have multiple collections, now it seems like you have one name index, and many Solr documents are added to this one index.

Ww: there is only one heterogenous Solr collection.. Here is the bit that flattens:

Hk: not clear on what a nested concept would look like, help to have an example. We don't use any nesting in VIVO.

Ww: (shows example of nesting) :

Huda: Can we take a few example SPARQL queries and start there? Just one type of entity. Does the resulting Solr index looks right? The nested documents is probably the biggest change.

Brian: If we took the existing SPARQL query document modify and extended it? William: Yes as long as….?

William: Does a document modifier 'know' what type of document it needs to make? Brian: No, because there aren't any separate base types.

Huda: Has to leave… as the host, the meeting ended :)

De: this nesting is not as deep as some other examples.

Ww: the nested documents are intentionally partial. Full document is fetched later and is swapped out for the partial bit.

Aw: so, what would be the next steps to integrate this with VIVO core?

Ww: need to decide if this is an acceptable shape of a Solr document that we want to work with?

De: so if the Solr document changes the graphql has to change as well.

Ww: yes, there are important implications. Need to document the instruction set for how documents are shaped

Also, William notes on VIVO Solr Indexing.

Actions

-