Infrastructure - Story #8082

Story # 8061 (New): develop queue-based processing system for the CN

implement SolrCloudClient to replace HttpService to allow concurrent updates of the solr index from differen machines

2017-04-25 22:47 - Rob Nahf

Status:	New	**Start date:**	2017-04-25
Priority:	Normal	**Due date:**	
Assignee:	Rob Nahf	**% Done:**	0%
Category:	d1_indexer	**Estimated time:**	0.00 hour
Target version:			
Story Points:			

Description

Based on Chris' advice:

Chris: So, to add my $.02, I think that yes, we can parallelize the indexing across CNs using a federated queue, and it's what we (somewhat desperately) need. One thing to consider, though, is how our clients handle conflicts.

As I understand it, the Solr server use the internal _version_ field to handle concurrent update requests on documents. If the client request has the same _version_ as the server, the request should succeed. However, if it doesn't, the expected behavior is for the server to return an HTTP 409 error, and the client is suppose to GET the latest document again and resend the update request. This may very well be baked into the SolrJ HttpSolrClient and the ConcurrentUpdateSolrClient (which apparently optimizes requests by consolidating multiple updates into a single HTTP request). However, our indexer code doesn't use the SolrJ client, but rather the home-grown HTTPService class that performs the update requests. A quick glance at that shows that it just logs all errors coming back from Solr, and doesn't handle _version_ mismatches. So, in moving toward concurrent clients, we might consider moving to the SolrJ optimized clients if they look like they handle mismatches gracefully.

Davev: and perhaps use a queue for feeding solr. Indexer tasks add processed docs to the queue, where a process using solrj sends to solr.

The CloudSolrClient seems to fit the bill...

https://community.hortonworks.com/questions/9611/concurrentupdatesolrclient-vs-cloudsolrclient-for.html

Related issues:

Related to Infrastructure - Story #8702: Indexing Refactor Strategy	**New**	**2018-09-24**

Associated revisions

Revision 18933 - 2017-09-14 18:07 - Rob Nahf

refs: #8082: Half-tested implementation of SolrJ-based client to the solr cores, with CloudSolrClient as the default implementation. Still need to test updates and deletes, tested querying. SolrJClientIT is an integration test for the class.

Revision 18933 - 2017-09-14 18:07 - Rob Nahf

refs: #8082: Half-tested implementation of SolrJ-based client to the solr cores, with CloudSolrClient as the default implementation. Still need to test updates and deletes, tested querying. SolrJClientIT is an integration test for the class.

Revision 18941 - 2017-09-27 21:10 - Rob Nahf

refs: #8082: Cleanup of SolrJClient - simplified constructors and created test-parser-context.xml for flexible implementation via Spring (where it should be). Added a testTypicalPackageIndex test, and test package documents.

Revision 18941 - 2017-09-27 21:10 - Rob Nahf

refs: #8082: Cleanup of SolrJClient - simplified constructors and created test-parser-context.xml for flexible implementation via Spring (where it

should be). Added a testTypicalPackageIndex test, and test package documents.

Revision 18942 - 2017-10-02 20:41 - Rob Nahf

refs: #8082, SolrJClient is set up to use the hard-commit-after-update that the current client does, to enable testing. configurable in the source code. Also only uses real-time-get if set. (RT get seems to take longer). Finished the simple package indexing test. Fixed a couple NPE errors in subprocessor, that arise if identifiers are looked up in Hz systemmetadata map and are not there, that is, when resourceMaps are indexed before their members.

Revision 18942 - 2017-10-02 20:41 - Rob Nahf

refs: #8082, SolrJClient is set up to use the hard-commit-after-update that the current client does, to enable testing. configurable in the source code. Also only uses real-time-get if set. (RT get seems to take longer). Finished the simple package indexing test. Fixed a couple NPE errors in subprocessor, that arise if identifiers are looked up in Hz systemmetadata map and are not there, that is, when resourceMaps are indexed before their members.

History

#1 - 2017-04-25 22:50 - Rob Nahf

- Description updated

#2 - 2017-04-25 22:51 - Rob Nahf

- Description updated

#3 - 2017-04-25 23:33 - Rob Nahf

- Description updated

#4 - 2017-04-26 04:56 - Rob Nahf

- Parent task deleted (#8081)

#5 - 2017-04-26 04:56 - Rob Nahf

- Parent task set to #8061

#6 - 2017-09-06 18:22 - Rob Nahf

regarding Dave's comment in the desxcription, see ConcurrentUpdateSolrClient
https://lucene.apache.org/solr/5_3_1/solr-solrj/org/apache/solr/client/solrj/impl/ConcurrentUpdateSolrClient.html

as well as this article on batching updates: https://lucidworks.com/2015/10/05/really-batch-updates-solr-2/

Both point us to the fact that batching updates reduces overhead and can speed up performance.

Need to check to see if optimistic concurrency is used by SolrJ be default. We'd need to at least put the *version* field in the updates we are submitting, I believe to enable this kind of concurrency control. This would be good because it will reduce race situations.

(Optimistic concurrency - "If first you don't succeed, try, try again")

For additional improvements, we also need to look at atomic updates, these can be combined with optimistic concurrency chris mentioned above: see http://yonik.com/solr/atomic-updates/

#7 - 2017-12-26 18:14 - Dave Vieglais

- Sprint set to Infrastructure backlog

#8 - 2018-09-24 15:29 - Rob Nahf

- Related to Story #8702: Indexing Refactor Strategy added