Infrastructure - Task #3726

Design strategy for dealing with data packages containing 100k objects in the search index.

2013-04-24 15:39 - Skye Roseboom

Status:	Rejected	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Dave Vieglais	**% Done:**	0%
Category:	dataone-cn-index	**Estimated time:**	0.00 hour
Target version:	CCI-2.4.0		
Milestone:	None	**Story Points:**	
Product Version:			

Description

Currently the search index is modeling the ORE data package relationships in three column - resourceMap, documents, documentedBy. These correspond to the 'aggregates', 'describes', 'describedBy' relations defined in the ORE document.

DataONE is currently deploying the search index using solr 3.6. This version does not allow partial updates - the entire record is updated/inserted and re-indexed. Solr does not provide the capability for specifying multiple records to update. This means that to enter the relationships defined in ORE documents: n+1 update/inserts are required to record the relationships in the search index. Each document in the index will need to be updated to record the resourceMap, documents, documentedBy relationships. This means to record a data set with 100k objects will take 100k updates to the solr index. This processing time to do this may become a performance issue.

Another issue is the use of a solr multivalued field (array) to model the 'documents' and 'documentedBy' relationships. In the case of large data packages, this field records the pid of each document in the relationship - potentially 100's of thousands of pids - in a large data set. Solr will attempt to store any number of items in the multivalued field but at some point, performance issues will arise.

Furthermore use of the ORE relationships becomes difficult when an object is in more than one data set. Since all the 'documents,documentedBy' relationships are stored in one field - users cannot easily determine which relationships in those fields correspond to which data packages.

The design and strategy of how DataONE and the search index presents relationships between objects needs some consideration of the best way to represent these relationships internally and how to present to the user.

Large datasets and how DataONE handles them will effect other clients/ITK tools - as they will need to determine how to display/present data package information and relationships for large data sets as well. For example - the download panel in OneMercury. Can it show 100k data package? Should it? Is this useful for users?

History

#1 - 2013-05-08 17:12 - Skye Roseboom

One possible solution for this problem would be a seperate solr index for recording 'triples' from resource maps. This would remove packaging data from the search index - allowing parallel updates to the search index.

The idea for using solr would be its ability to scale up to dealing with millions of documents - and possibly shard (solr cloud) in the future - as needed.

Downside is support for sparql query syntax. DataONE could fake or just expose its own data service / REST endpoints in front of the solr triple store....

#2 - 2013-05-09 18:59 - Rob Nahf

For the display of contents of a package, perhaps creating an xslt/stylesheet for the RDF itself would be helpful, by providing at least a default html presentation of the content, and resolvable links to the objects themselves.

The ONEMercury download display works well enough for small packages, but after a certain number of items, is impractical. Maybe something like this could take over.

For other issues like index update performance, and search (i.e.: q=documents:{pid}) , I agree that a triple store is probably a better solution. despite it having the effect of decentralizing query functions.

#3 - 2013-05-16 16:59 - Skye Roseboom

- Parent task set to #3760

#4 - 2017-03-28 16:10 - Dave Vieglais

- Parent task deleted (#3760)

- Milestone set to None

- Category set to dataone-cn-index

- Project changed from CN Index to Infrastructure

- Target version set to CCI-2.4.0

#5 - 2019-04-30 22:30 - Dave Vieglais

- Status changed from New to Rejected

This item is covered with the refactoring of the indexing process to leverage record partial updates in solr 5+