Infrastructure - Decision #8693

Support Google Dataset Search on search.dataone.org via partial server side rendering

2018-09-07 00:16 - Bryce Mecum

Status:	In Progress	**Start date:**	2018-09-06
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	30%
Category:		**Estimated time:**	0.00 hour
Target version:			
Milestone:	None		

Description

Background

Yesterday, Google Dataset Search launched. We previoiusly attempted to make MetacatUI (and by extension, DataONE Search) compatible with it by injecting Schema.org JSON-LD into appropriate pages. During development and testing, we checked our compatibility with the upcoming Google Dataset Search using Google's Structured Data Testing Tool. During development, this was all working fine and the feature appeared to be compatible but, after launching the feature on search.dataone.org, behavior changed on Google's end making it so Google no longer saw this JSON-LD. The reason for this is likely that, because MetacatUI follows a single page application architecture and we inject the JSON-LD on the client side, Google's JSON-LD crawler only saw what was sent from the server (a nearly empty index.html) and not our full page (with JSON-LD). I was able to test this theory and, while Google's crawler does execute JavaScript, it limits execution to about or exactly five seconds and MetacatUI *usually* doesn't finish injecting JSON-LD and rendering all content until after that timeout.

Potential paths forward to get DataONE Search compatible with Google's Dataset Search include (none of which are mutually exclusive):

1. The assets that make up MetacatUI and the asset loading strategies could be optimized: https://github.com/NCEAS/metacatui/issues/224
2. Move the code (and any dependencies) that injects JSON-LD further up in the app boot so that Google sees it
3. Inject the appropriate JSON-LD on the server side to guarantee that Google sees it (originally Matt Jones' idea!)

(1) is being worked on for sure, and (2) may not be needed if (1) is successful. I want to talk about option (3) because:

- It's a quicker solution (I already have something working) which would help get us involved in the project faster
- It paves the way for future features and/or improvements to MetacatUI (we could be rendering more on the server side than just JSON-LD, like other metadata, more page content, etc)

What I did

To test this idea, I modified a previous project which is just a simple Node (Express.js) app that hosts MetacatUI by intercepting every request and serving the appropriate asset. In injects Schema.org JSON-LD, when appropriate, by querying the CN Solr index before sending MetacatUI's index.html to the client. Code is here and its deployed here. View source on any /view/... pages and you'll see a minimal Schema.org/Dataset description in the head. More properties can be added later. I did it quick and dirty: The app pre-loads MetacatUI's index.html as a String at app boot and injects the JSON-LD into it. No templating language or other magic.

Things to address

- How do we feel abouts switching from hosting MetacatUI via Apache (simple, bullet proof) to a Node based deployment just to support this feature (new territory, at least for me)?
- If we do switch, we'd want to make really sure the Node app doesn't have weird failure cases where it doesn't return index.html (e.g., when Solr is down, or slow). The app needs to return index.html (and every other static asset) on every request and do it very fast and we should decide what the cutoff is so that it doesn't hold up app boot if Solr is slow/down.
- Can this type of deployment easily be integrated with CN buildouts? I've deployed Node apps before by fronting them with Apache/nginx (via reverse proxy) and then keeping the node process up with Upstart
- Is this performant enough for DataONE? I think my implementation is non-blocking but I'm not a Node expert so we'd want to code review and probably benchmark
- We could wait on (1) and stick with our current deployment strategy

Other notes

Unrelated to the Google Dataset Search issue but related to Google's crawling for Google Search, we've also identified:

- That the Metacat View Service is often unreasonably slow: https://github.com/NCEAS/metacat/issues/1234 and are planning to figure out why
- That we can and should make use of sitemaps to help Google crawl our pages: https://github.com/NCEAS/metacat/issues/1263

History

#1 - 2018-09-08 00:35 - Dave Vieglais

Optimizing search UI for rendering performance is a good move, however the application is fundamentally orthogonal to LOD in that the server response to a client request is an application that must be executed to retrieve a resource rather than the requested resource.

Server side processing is the alternative, and switching to nodejs is one path towards such a refactor though other technologies (e.g. java or python) would work just as well, though this is what nodejs does work quite well for.

#2 - 2018-09-10 21:37 - Bryce Mecum

Thanks Dave. Two thoughts:

1. Re: "however the application is fundamentally orthogonal to LOD". If I understand you, I think it'd say that MetacatUI and JSON-LD are only orthogonal if you don't consider that Google only wants to see JSON-LD in HTTP response bodies from web applications and static sites and not in a standalone fashion. I don' think I fully understood your point though.
2. nodejs felt like a natural choice since we've already written this code in JavaScript, Node is probably performant enough for our use case, and I have written Node before. I don't think implementing this in another lang would be a pain though. I do think Node is a lower friction path towards adding more features such as partial SSR and client side hydration.

#3 - 2018-09-11 01:20 - Dave Vieglais

wrt #1, it's very simple - when requesting a resource with a URL, the resource should be returned, which in this case is reasonably expected to be the requested item. Instead, MetacatUI returns an application that must then be executed to retrieve the requested resource. While technically correct from a HTTP semantics point of view (since the URL does point to an application), it does make it far more complicated for a client to actually get and inspect the listed resource, especially when the goal is to support LOD.

#4 - 2018-09-11 13:16 - Bryce Mecum

Ah, yes. Thanks for clarifying.

#5 - 2019-06-06 23:53 - Bryce Mecum

- % Done changed from 0 to 30

- Priority changed from Normal to Low

- Status changed from New to In Progress

Google staff have indicated we only need to send them a robots.txt that points to our sitemaps for them to begin crawling so we're going to try that first (See https://redmine.dataone.org/issues/8817) and come back to this if necessary.