EarthCube Project 419

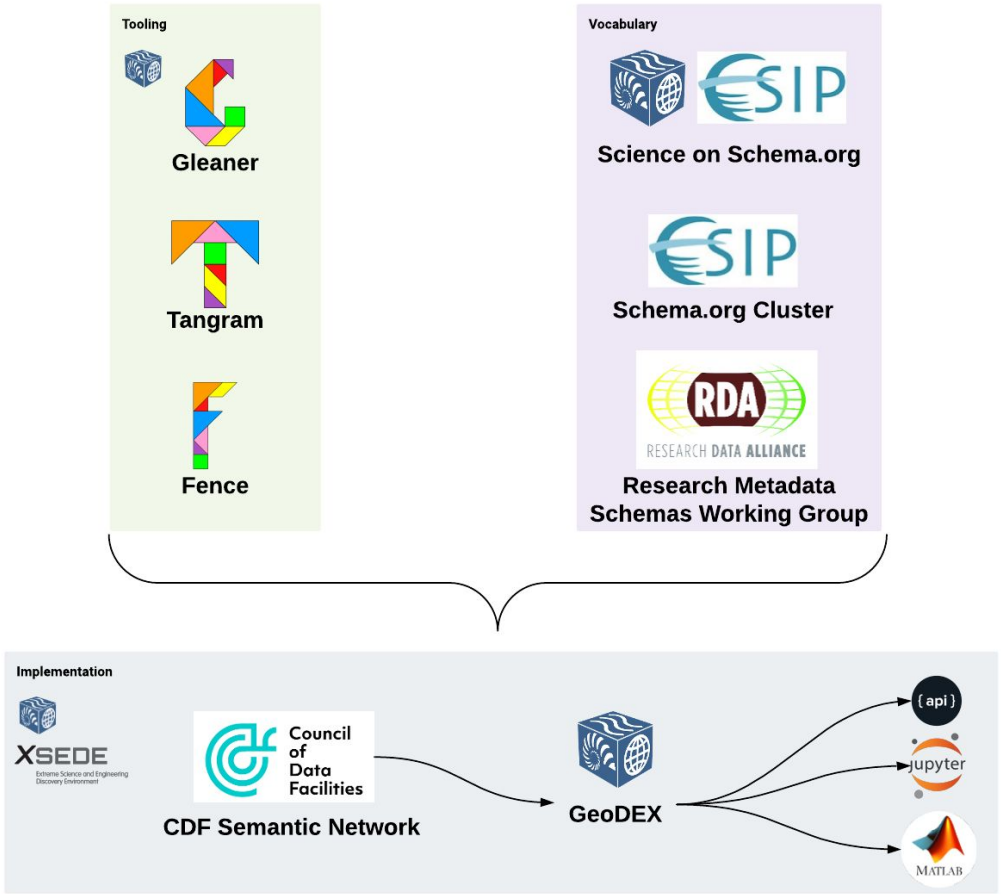
Doug Fils (dfils@oceanleadership.org), Adam Shepherd (ashepherd@whoi.edu)

Summary

Project 419 developed a set of tools and vocabulary to support the structured data on the web lifecycle. These included:

- Gleaner: A tool to harvest and index a set of resources from a community. The resources are defined by provider sitemaps pointing to web resources containing data graphs of structured data in a JSON-LD encoding.
- Tangram: A service that support Gleaner by providing a means to validate the structure of the data graphs Gleaner reads using a community developed and maintained validation or shape graph.
- Fence: An online tool for developers using elements of Gleaner and invoking Tangram to provide an interface to allow developers to inspect and validate the data graphs during the adoption phase.

These tools and vocabulary provide a foundation for a data workflow.



This data flow can be divided into three main sections for discussion. These being:

- Tooling: Comprised of Gleaner, Tangram and Fence
- Vocabulary: Comprised of the science on schema work along with the extension work for encoding relations and the shape graphs for validation
- Implementation: Where products of the tooling, vocabulary generating indexes and networks support queries and interfaces.

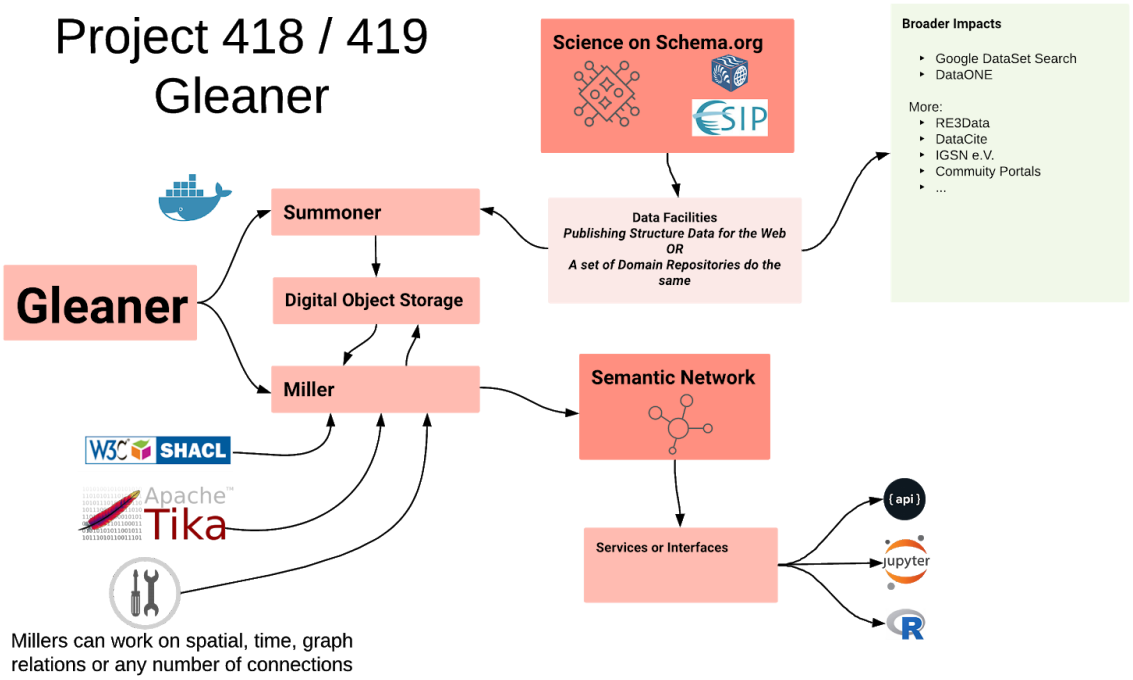
Special note is given by the authors at this point to the real heros, the data providers. The providers who develop the internal workflows to map their data and metadata holdings to these representations and onto the open web architecture enable all this. These material feeds large scale implementations of this pattern, like Google DataSet Search (https://toolbox.google.com/datasetsearch). They also support the development of smaller focused semantic networks like the CDF Semantic Network to be discussed here.

Tooling

The tooling has evolved from the P418 work to be fully deployable in a containerized system and use configuration files. The goal of this work for Gleaner is to allow a larger audience of potential users to be reached. The technical level is targeted at people who can install and run the Docker runtime on their computers and have a basic understanding of running command line and script based applications.

The Tangram and Fence packages run on-line and can be deployed to any container aware environment. These systems were developed on the NSF XSEDE system and are currently running on Google Cloud, though as containers could be deployed to Azure or AWS as well.

Gleaner (https://gleaner.io)



https://github.com/earthcubearchitecture-project418/gleaner

Gleaner is the structured data on the web indexing tool developed as part of the NSF EarthCube Project 418. Gleaner's focus is on collecting JSON-LD encoding data graphs about data resources and services. Gleaner can then process and generate a semantic network based on a list of providers. These can then be used to build services or connected with other networks.

Gleaner code is available at GitHub. The approach to the code resulted in two seperate sections focused on different concerns.

Summoner

The Summoner section of Gleaner is focused on reading the sitemap and accessing the resources. Summoner can deal with both static pages containing JSON-LD and also dynamically generated pages where the JSON-LD is loaded after page load by Javascript calls. Currently it can not inspect for this though, so the setting must be set in the configuration file.

Summoner will also validate that the JSON-LD is well formed and can be encoded into other RDF data model representations like n-quads for easier loading into a range a triple stores. It will separate the presented data graphs into a collection of "good" and "bad" triples. Here "bad" means triples that violate some aspect of the RDF standard and need attention by the facility. These "bad" triples become part of the report back to the facility on their data graph to allow them to address or question the categorization.

Miller

The Miller section of Gleaner represents the extensible workflow portion of Gleaner. It is where data graphs can be post processed. Where Summoner collections information, Miller converts the information into a usable product. Some examples of these are

Graph Miller

This is the default miller and generates the RDF graphs. It's a simple workflow that converts the JSON-LD graphs into n-quads. This miller now also makes sure the JSON-LD is well formed and that the RDF triples are following proper structure as defined by the W3C guidelines for RDF.

Spatial Miller

This was a miller developed during the Project 418. It extracted the schema.org based spatial data and converted it to GeoJSON. During P419 this was modified to use JSON-LD framing and converts the schema.org based geospatial data into geometry primitives. From these geometries then any of GeoJSON, WKT or KML can be generated. Currently we are working to integrate this miller with the WKT output to generate triples that work with GeoSPARQL queries.

SHACL Miller

The SHACL miller can use W3C SHACL shape graphs to validate shapes in the data graphs published by providers. Currently we have shapes that define the Google Structured Data for Developers guidance for type Dataset. This includes the recommended and required types and properties. This system can use any shape graph as input and newer shape graphs are under development.

Full text (Tika) Miller

The Tika miller uses the distribution URL present in the JSON-LD based data graphs to access the actual data files. These can then be processed through the Apache Tika software to allow the extracted text from the files to be placed in the graph to aid in full text indexing if desired.

Since this is a rather intensive process it has only been done on a couple repositories and would not be a default operation we would recommend. Such a process and event should be discussed and agreed on with the data providers so as not to over stress servers.

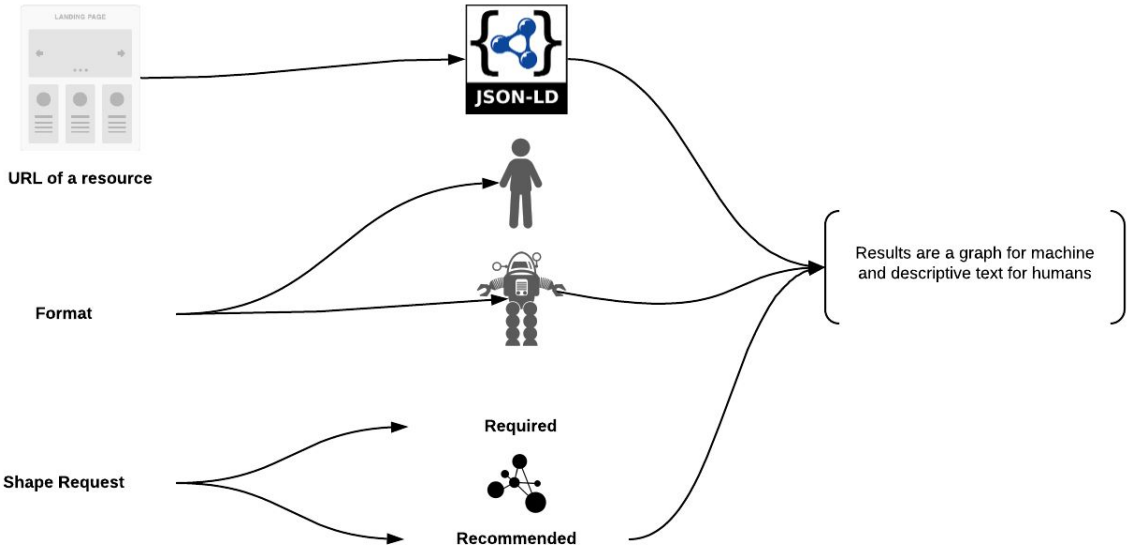
Future options like NER

Extracting text from the JSON-LD or data files also opens the possibility for future millers leveraging Natural Language Processing (NLP) and Named Entity Resolution (NER). Some initial work has been done on NLP and NER approaches but, at the time of this report, these are only some simple exploratory work.

Tangram (https://tangram.gleaner.io)

The Tangram services is a web service wrapper around the pySHACL package. It allows you to send in JSON-LD data graphs to test against a Turtle (ttl) encoded shape graph. See the GitHub Repo for some more details on using this service. The shape graph work is taking place at geoshapes.org.

- https://tangram.gleaner.io (not a web ui, but a service call only)
    - For notes reference: https://github.com/geoschemas-org/geoshapes/tree/master/src/tangram



Validation (https://geoshapes.org)

Gleaner is now a form and validation checking tool. That is, it can now inform facilities if their data graphs (JSON-LD) are well formed and also that is is valid against a set of shape graphs. We have defined shape graphs for Google required and recommended for the data search tool based on their developer publications.

- Reference: http://geoshapes.org for the shape graphs and documentation

Fence (https://fence.gleaner.io)

Fence is a tool designed to allow people to evaluate structured data graphs against various validations, tools and interface components. It is a work in progress at this time. See the GitHub Repo for some more details on using this service.

- Fence: Reference https://fence.gleaner.io (new version not deployed, but this one works)
    - A tool for developers to inspect their data graphs and validate them against the SHACL shape graphs
    - Example runs of the earlier version can be done there.. Fence calls Tangram for validation.

Architecture

As mentioned previously the Gleaner system and the derivative Tangram and Fence tools are all containers based on Docker. All the development work was done on the NSF XSEDE resources, specifically JetStream.

Tangram and Fence have been migrated to Google Cloud, mostly as a means to test some of the commercial cloud approaches and ensure work on XSEDE ported well to them. Both Tangram and Fence are hosted in the Google Cloud Run offering.

The Gleaner work is also done with the existence of the NITRD Open Knowledge Network in mind. The CDF Semantic Network (referenced below) was done to being developing a workflow for Gleaner that resulted in a product that could contribute to the domains of practice contributing to this work.

Implementation

The output of Gleaner on the EarthCube Council of Data Facilities (CDF) was used in a test implementation. The resulting semantic network was loaded into a triplestore (Blazegraph and later Jena/Fuseki) and used to resolve service calls. These APIs are defined in swagger [1] and power the services exposed by the container built from the P418 Services code base [2]. To check for performance and regresion a set of testing scripts was developed using the python Tavern system. Wrapping this together, a simple text search interface for this can be seen at https://geodex.org.

[1] http://geodex.org/swagger-ui/
[2] https://github.com/earthcubearchitecture-project418/services

Scale

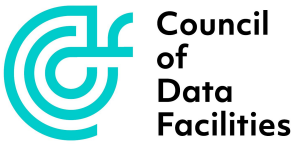
Two events are driving desire for improved scaling in Gleaner. Previously Gleaner only had to do with scale in the 10s of thousands with a total resource index in the 100K range. Interest in Gleaner by groups dealing with 10s and 100s of millions drive the need to improve the scaling.

The original Gleaner code was memory bound in such a way that scaling to millions was not feasible except on very large hardware. A new memory mapped disk based key value approach was implemented and currently Gleaner is indexing into the million range reliably.

The system is also Docker based so can take advantage of container orchestration scaling. Additionally, the code base is written entirely in the Go programming language which is used to build highly concurrent execution elements in Gleaner. This allows the code to leverage multi-core machines well.

Semantic Network (domain knowledge graph)

Gleaner generates output in a format that can be loaded into triplestores and scripts for loading Gleaner output directly into triplestores have been developed. The output of Gleaner is effectively a semantic network for the group of providers indexed. It forms a semantic network or domain level knowledge graph (when augmented with the ontologies used). An example of this is from index members of the EarthCube Council of Data Facilities (CDF) can be found at:



https://github.com/earthcubearchitecture-project418/CDFSemanticNetwork .

This allows anyone in the community to access and use the network to provide value add products or insight.

Vocabulary

The vocabulary work as done as a separate contract with Woods Hole Oceanographic Institute and the report on that work is available as a separate document. In particular reference the report at: https://github.com/ESIPFed/science-on-schema.org

Deliverable 1
Per the Statement of Work, WHOI reviewed schema.org's representation of time for Dataset resources. This representation supports date and time formatted in ISO 8601 but neglects other time formats relevant in the geosciences such as fuzzy time and duration intervals. W3C OWL-Time Ontology (https://www.w3.org/TR/owl-time/) supports both these concepts by defining a temporal reference system that lays the groundwork for supporting concepts of time outside the Julian calendar. First, it defines a generic temporal entity from which specific concepts of time derive = namely instants in time and intervals of time.

Importing OWL Time for use into schema.org required a small extension vocabulary to be developed that links the Dataset concept to this generic temporal entity. This link, through RDF class relations, supports use of instants and intervals of time as they subclass, or inherit, from the generic temporal entity. This small extension is published at https://geoschemas.org/contexts/temporal.jsonld

Use of this Dataset temporal coverage extension is described in the Github repository: https://github.com/earthcubearchitecture-project418/p419voctemporal which provides a high level explanation of OWL Time along with examples for using the extension within schema.org markup. These examples cover describing intervals, instants in time, including fuzzy time such as Geologic time scales, durations of time, for streaming datasets, and finally seasonal datasets.

The Earth Science Information Partners (ESIP) Schema.org Cluster will review this extension and guideline documentation for inclusion in the guidelines at http://science-on-schema.org

Deliverable 2

Per the Statement of Work, WHOI assisted with Focus Areas 1,3,4,5 of the subaward to develop guidance documents on publishing Data Services with Schema.org. WHOI identified 2 use cases for describing data services - smart hand-offs and deep queries. Smart Hand-offs describe the ability for a data service to represent in the schema.org markup the various inputs it can receive to enact a service. A harvesting engine can then utilize this information to jump into the service by passing along inputs relevant to that service's capability. This capability is described by Doug Newman and the NASA Giovanni application (https://ntrs.nasa.gov/search.jsp?R=20190000091). The second use case, deep query, involves a harvester, querying a data service endpoint, retrieving results, parsing those results, and presenting them within a larger search result set made by a search user. While this capability does not scale to the large harvesters such as Google, Bing, and Yahoo!, this capability might be possible for smaller scoped domains such as the geosciences.

All research into supporting the description of data services in schema.org is published at Github: https://github.com/earthcubearchitecture-project418/p419dcatservices . To support the description of Data Services, WHOI researched DCAT Version 2 Working Group (https://w3c.github.io/dxwg/dcat/) and published journal articles from commercial industry on smart hand-offs (https://doi.org/10.1016/j.procs.2018.09.025). Previous guidance documentation at http://science-on-schema.org - Dataset Guidance - Accessing Data through a Service Endpoint detailed how to describe a data service endpoint for supporting deep query, and new guidance documentation was developed to describe web APIs and their service documentation using the Schema.org WebAPI class.

Additionally, example schema.org was developed to describe the Earthcube CHORDS project and its streaming data services: https://github.com/earthcubearchitecture-project418/p419dcatservices/tree/master/CHORDS

Conclusion

The authors would like to acknowledge the support and help of the ESIP and RDA community in all this work. Discussions, feedback, presentation opportunities and more were highly valuable to the development and evolution of these products.

In particular the EarthCube Council of Data Facilities, ESIP Semantic Technologies Committee, ESIP Schema.org cluster and the RDA Research Metadata Working Group communities provided a large amount of feedback to this work.

All the work is open and available as git repositories hosted at GitHub and at the web sites listed in this report. A large amount of interest has developed around this work from many places and the authors look forward to both continuing to contribute and also encouraging others to embrace and contribute to this work.

References:

The following are some references used in this report. They are provided for completeness.

Tooling

- Gleaner: Reference https://gleaner.io
    - Can now be deployed entirely in Docker. User dependency is a working Docker install only. Can be deployed locally or in the cloud. (a video of how to use this is coming... also we have 3 external groups interested in using Gleaner now)
- Tangram: Reference https://tangram.gleaner.io (not a web ui, but a service call only)
    - For notes reference: https://github.com/geoschemas-org/geoshapes/tree/master/src/tangram
- Fence: Reference https://fence.gleaner.io (new version not deployed, but this one works)
    - A tool for developers to inspect their data graphs and validate them against the SHACL shape graphs
    - Example runs of the earlier version can be done there.. Fence calls Tangram for validation.

Validation (Shape graphs)

Gleaner is now a form and validation checking tool. That is, it can now inform facilities if their data graphs (JSON-LD) is well formed and also that is is valid against a set of shape graphs. We have defined shape graphs for Google required and recommended for the data search tool based on their developer publications.

- Reference: http://geoshapes.org for the shape graphs and documentation

Semantic Network (domain knowledge graph)

Gleaner generates output in a format that can be loaded into triplestores. Scripts for loading Gleaner output directly into triplestores has been developed. The output of Gleaner is effectively a semantic network for the group of providers indexed. It forms a semantic network or domain level knowledge graph (when augmented with the ontologies used). An example of this is at: https://github.com/earthcubearchitecture-project418/CDFSemanticNetwork . This allow anyone in the community to access and use the network to provide value add products or insight.

Temporal

- Open-ended intervals for unbounded temporal coverages (schema-org issue:1365)
    - Streaming data, ongoing research w/o known end-date
- **OWL-Time work**
    - https://github.com/earthcubearchitecture-project418/p419voctemporal
    - Granularity of dates/times: intervals, durations, seasonality (OWL-Time issue: 1140)
    - Completed Time Work: Sept 9, 2019
        - Context: http://schema.geoschemas.org/contexts/temporal
        - Documentation: https://github.com/earthcubearchitecture-project418/p419voctemporal/blob/master/README.md
    - Future Work:
        - ESIP schema.org Cluster review & approval
        - Move to geoschemas.org (website for external extensions schema.org)

Services

- Vetting Data Services with W3C (issue:4)
- CHORDS implementation
- Initial discussions about IRIS w. Rob Casey - describing Datasets & Data Services
    - IRIS preference is emphasis on DataSets but being able to indicate the supporting Data Services as sub-elements. From the ESIP meeting, it looked like things were heading that way.
        - https://docs.google.com/document/d/1aguPf1KpWyIk5LGt__wfl7EILMDJnFCdJauJAd2U_Fk/edit#heading=h.g2qlyiq1auj
    - IRIS will want to separate its DataSets by the discrete seismic networks registered with the FDSN: http://www.fdsn.org/networks/
- Updated README with axioms: Sept 7, 2019
    - **Smart Hand-off** : harvester knows how to pass your query off to the Service
        - Works now (Google "hotel reservation" => form to query hotels...)

- **Deep Query**: harvester knowing how to query and display results
  - Might not scale to a global harvester
  - But, something scoped to a smaller subset of the web might be able to do
    - Example: http://erddap.com/