Serverless GEO labels for the Semantic Sensor Web

Anika Graupner

Institute for Geoinformatics, University of Münster, Münster, Germany
a_grau05@uni-muenster.de

Daniel Nüst [1]   ⓘ

Institute for Geoinformatics, University of Münster, Münster, Germany
daniel.nuest@uni-muenster.de

──── **Abstract** ────

With the increasing amount of sensor data available online, it is becoming more difficult for users to identify useful datasets. Semantic Web technologies can improve such discovery via meaningful ontologies, but the decision of whether a dataset is suitable remains with the users. Users can be aided in this process through the GEO label, which provides a visual summary of the standardised metadata. However, the GEO label is not yet available for the Semantic Sensor Web. This work presents novel rules for deriving the information for the GEO label's multiple facets, such as user feedback or quality information, based on the Semantic Sensor Network Ontology and related ontologies. Thereby, this work enhances an existing implementation of the GEO label API to generate labels for resources of the Semantic Sensor Web. Further, the prototype is deployed to serverless cloud infrastructures. We find that serverless GEO label generation is capable of handling two evaluation scenarios for concurrent users and burst generation. Nonetheless, more real-world semantic sensor descriptions, an analysis of requirements for GEO label facets specific to the Semantic Sensor Web, and an integration into large-scale discovery platforms are needed.

───────────────

[1] Corresponding author.
     All URLs in this document were last checked on February 19, 2020.

1   Introduction

The amount of sensor data captured and accessible today is ever increasing, not least because of the numerous sensing devices that are part of the Internet of Things, Smart Cities, and the newest Earth observation satellites. The Group on Earth Observation's (GEO) Global Earth Observation System of Systems (GEOSS) [6] has attempted to make near real-time environmental data and processing available to users in the form of a Spatial Data Infrastructure (SDI) based on OGC Web Services and standards (cf. [14]. However, the complexity of the task and the continued growth of sensor data means that this system is, and probably will remain, an evolving work in progress. To improve user recognition of geospatial datasets, to promote trust in datasets, and to assist users in the discovery of suitable datasets, the GEO label was developed [18]. The original GEO label design, as described by Lush [18], comprises several *facets* to convey the most relevant information to users. The label, as a graphical representation for individual datasets in GEOSS, allows users to compare complex metadata as well as objective and subjective quality information to make an informed decision when selecting a dataset [19]. To integrate the GEO label with geospatial catalogues and applications, the generation of the GEO label was encapsulated in a RESTful Web API, the *GEO label API*[2]. This API creates labels in SVG format [8] and other image and machine-readable formats for provided metadata documents or for links to online documents; SVG is the primary format because of its flexibility for scaling and because it enables interactivity, such as pop-ups or links for parts of the image. To transfer the label's usefulness to standardised sensor data, Nüst et al. [24] extended the GEO label for the OGC Sensor Web Enablement [3] (SWE).

However, the GEO label is not used in production today, and its potential for improving sensor data discovery is subsequently untapped. This can be traced back to three gaps, namely a gap in (1) application of the GEO label to additional concepts and implementations for interoperable data exchange, such as the Semantic Web[3], to grow the supported metadata sources and increase coverage, (2) implementation of ready-to-use and scalable platforms for integrating the label into existing services (APIs, portals) and their requirements, and (3) adoption of the label by operators of online sensor data portals.

Regarding the first gap, we propose to extend the GEO label to include metadata from the Semantic Sensor Web (SSW, [25]), which is an important infrastructure for sensor data complementing OGC SWE. Janowicz et al. [14] describe how SDIs can be enhanced with a transparent mapping to the Semantic Web, and the SSW connects the Semantic Web with the OGC SWE suite of standards. The SSW offers a framework for enabling interoperability and meaningful data integration, processing, and reasoning. In the SSW, the metadata captured by ontologies provide a promising source of information for the GEO label, because the information is meaningful and can be drawn from various linked resources. In turn, the GEO label has the potential to improve data discovery in the vastness of geospatial sensor datasets, whereby characteristics such as the lack of a singular inventory or the dynamic structure of sensor webs represent key challenges [16]. Previous work [1, 15, 5] uses specialised ontologies or queries to answer the discovery challenges in OGC SWE and the SSW, but no existing approaches use a visual badge or label for improved data discovery in the Semantic Web.

Regarding the second gap, the generation of labels for data portals must serve two

---

[2] `https://geolabel.net/api.html`
[3] `https://en.wikipedia.org/wiki/Semantic_Web`

different approaches depending on the platform. Demand is either small, intermittent, and unpredictable, if labels are generated on demand with a discontinuous workload depending on users in interactive sessions, or demand is schedulable in isolated but large bulk events, if labels are generated and stored regularly for all available metadata. To serve both scenarios, we propose to deploy the GEO label API to cloud computing infrastructures.

Finally, regarding the third gap, tackling such organisational or strategic issues is out of scope for this work. Nevertheless, closing the former two gaps indirectly helps stakeholders and operators of public or open infrastructures to adopt the label in practice.

The **main contributions** of this work are (1) creating a mapping between metadata fields of the Semantic Sensor Web and the GEO label facets, (2) implementing a prototype of this mapping which conforms to the GEO label API, and (3) evaluating the prototype in serverless computing infrastructures with respect to intermittent and bulk generation of labels. In the remainder of this work, we first identify suitable sources of information in ontologies of the SSW and related ontologies. Then we evaluate existing GEO label API implementations and different cloud computing providers to identify suitable base software and cloud platforms for a prototypical implementation. Finally, we evaluate the prototype's performance. See the *Supplement* section for information about the software prototype, the test data used, and online deployments of the prototype.

2   GEO label for the Semantic Sensor Web

The SSW's main ontology is the *Semantic Sensor Network Ontology* (SSN, [12]). To create a mapping between the SSN and the GEO label, we first evaluated the modular SSN for suitable fields which can provide meaningful information for the different GEO label facets. This evaluation included SSN's core ontology SOSA (Sensor, Observation, Sample and Actuator) and the SSN's aligned modules, such as the Provenance Interchange Ontology (PROV-O) [17]. Each of the over 50 classes and properties of SSN and SOSA and their aligned modules (100+ classes an properties) was checked one by one against the eight facets of the GEO label. Next, we extended the search to include ontologies often used in conjunction with SSN starting from the SSN specification's examples[4]. From those, we adopted generic properties for names and descriptions, e.g., using the Friend of a Friend ontology (FOAF) [4]. Finally, we looked more broadly for ontologies on topics with a relation to until then not covered facets using the Linked Open Vocabularies catalogue[5]. This search lead eventually led to the usage of the Dataset Usage Vocabulary (DUV) [10] and the Bibliographic Reference Ontology (BiRO) [9].

For example, for the facet *Producer Comments* is set to available if a document contains an `rdfs:comment`, because we can assume that such a comment stems from an entity involved in the creation of the metadata record, for the facet *Compliance with Standards*, the mapping checks if one of the used URIs contains `w3.org` and thereby denotes usage of a vocabulary that underwent a development under the auspices of the Word Wide Web Consortium (W3C), while for the mapping *User Feedback* an observation, `sosa:Observation`, must be connected to a `duv:UserFeedback` based on the `duv:hasUserFeedback` property. However, not all mappings are so open respectively direct or simple and allow different options. For example, for the facet *Producer Profile* an SSNO class such as `sosa:Sensor` can be connected to a `prov:Agent` using either `prov:wasAttributedTo` or `prov:wasAssociatedWith` and the

---

[4] `https://www.w3.org/TR/vocab-ssn/#examples`
[5] `https://lov.linkeddata.es/dataset/lov/`

respective PROV subclasses, and for the facet *Lineage Information* any of the relations `ssn:implements`, `ssn:implementedBy`, or `sosa:usedProcedure` can connect a sensing system with its procedure documentation.

Table 1 summarises the result of the manual process and briefly explains the reasoning behind the chosen mapping. See Section 3.2 for the full details on the mapping and the technical realisation. The table shows the ontologies, classes, and properties we identified as suitable sources for the GEO label's facets. The used ontologies and prefixes are listed in Table 2. Note that we did not add new alignments between the SSN and other ontologies, as that is beyond the scope of this work.

3   Serverless GEO label Generation

3.1   Serverless Computing

Serverless computing allows developers to deploy custom code in a shared infrastructure [2], whereby the application is maintained in a scalable way by a platform provider. The automated scaling enables both handling of large spikes of high demand and reducing costs when there is little or no demand. These properties make serverless computing a good fit for the GEO label generation usage scenarios. The GEO label generation can be deployed to a serverless infrastructure quite easily, i.e., without a complex setup including multiple services or a database, because each generation of a label is a relatively small, stateless, atomic operation. The creation of a label externally only relies on the metadata sources for which a label is requested. However, depending on the usage scenario, requests for labels can be erratic and unpredictable. To demonstrate applicability of the prototype the evaluations were conducted within the free tier of the following service providers: Google Cloud Run[6] (GCR) and Amazon Web Services (AWS) Lambda[7]. A comparison of the costs, while relevant for potential operators, is out of scope for this work.

3.2   GEO label API Implementation

The GEO label API is implemented in two software projects, one in Java and one in PHP[8]. In this work, the Java-based implementation is used because PHP is not supported by the serverless computing providers and the PHP project is no longer maintained. The rendering of the GEO label is based on an SVG template file. Labels are generated using the template file according to XPath expressions [7], which detect the presence of certain elements in a provided XML document. To use XPath, the RDF graph must be serialised in RDF/XML. Both implementations support a bespoke JSON-based configuration file format, which allows one to update the rules for transformations of metadata documents to labels without changes to the source code and to deploy these updates to GEO label API instances without updating the installation. To realise the conceptual mapping described above, we created a new transformation file[9]. The file is activated when the implementation is provided as an RDF/XML document, i.e., if the XPath `boolean(/*[local-name()='RDF'])` testing the document's root element evaluates to true. Of note, the implementation of hoverover and drilldown features lies beyond the scope of the proof-of-concept implementation.

---

[6] `https://cloud.google.com/run/`
[7] `https://aws.amazon.com/lambda/`
[8] `https://geolabel.net/implementations.html`
[9] See transformation file source at `https://github.com/nuest/GEO-label-java/blob/master/server/src/main/resources/transformations/transformerSSNO.json`.

Table 1 GEO label facets' data sources in the Semantic Sensor Web

Facet	Availability & hoverover text	XPath[1]
Producer profile	One SSN object (e.g., sosa:Sensor) has a responsible prov:Person or prov:Organization associated using prov:wasAttributedTo or prov:wasAssociatedWith; hoverover shows name(s) using foaf:name.	//*[rdf:Description[rdf:about= //prov:wasAssociatedWith/rdf:resource or rdf:about= //prov:wasAttributedTo/rdf:resource]/rdf:type[rdf:resource= "http://www.w3.org/ns/prov#Organization" or .../prov#Person"]] (partial expression)
Producer comments	A comment provided *within* a sensor description is coming from the producer; hoverover shows count and excerpt of comments.	//*[rdfs:comment]
Lineage information	At least one usage of sosa:Procedure, either directly, or for an SSN object via one of ssn:implements, ssn:implementedBy, or sosa:usedProcedure; hoverover shows number and type of a procedures' ssn:Input and ssn:Output	//*[ssn:implements or ssn:implementedBy or sosa:usedProcedure or sosa:Procedure or rdf:resource= 'http://www.w3.org/ns/sosa/Procedure']
Standards compliance	If URIs include w3.org, at least one ontology is by the W3C as the authority for the Semantic Web.	//*[contains(*, 'w3.org')]
Quality information	An ssn:System has (a) the property ssn-system:qualityOfObservation or (b) at least one property detailing a sensor's capabilities (e.g., accuracy, battery lifetime) identified via the relations ssn-system:hasSystemProperty, ssn-system:hasOperatingProperty, or ssn-system:hasSurvivalProperty; hoverover shows name(s) of available properties.	//*[ssn-system:hasSystemProperty or ssn-system:hasOperatingProperty or ssn-system:hasSurvivalProperty or ssn-system:qualityOfObservation]
User feedback	A SSN object has a property duv:hasUserFeedback connecting it with a duv:UserFeedback or duv:RatingFeedback; hoverover shows counts of feedbacks and an statistics of categorical ratings.	//*[duv:hasFeedback/duv:UserFeedback[not(prov:qualifiedAssociation)] or duv:hasFeedback/duv:RatingFeedback] (partial expression)
Expert reviews	A user feedback (see above) is qualified with a relation prov:qualifiedAssociation describing the author's role with prov:hadRole as an expert, a role that is not specified within PROV-O but can be the literal expert; for hoverover see *User feedback*	//*[duv:hasFeedback/duv:UserFeedback/ prov:qualifiedAssociation/ prov:Association/prov:hadRole/prov:Role[contains(rdf:about, 'expert')]] (partial expression)
Citations information	An SSN object, e.g., sosa:Observation, has a property biro:isReferencedBy pointing to a piece of literature; hoverover shows count of references.	//*[biro:isReferencedBy]

1) Wrapping boolean(..) statements omitted; partial expressions show only one of multiple allowed statements combined with or; the full mapping configuration is available at .

■ **Table 2** Used ontologies and vocabularies with their prefixes, and namespaces.

Ontology/vocabulary	Prefix	Namespace
Resource Description Framework	`rdf`	`http://www.w3.org/1999/02/` `22-rdf-syntax-ns#`
Resource Description Framework Schema	`rdfs`	`http://www.w3.org/2000/01/rdf-schema#`
Semantic Sensor Network Ontology	`ssn`	`http://www.w3.org/ns/ssn/`
Sensor, Observation, Sampling and Actuator	`sosa`	`http://www.w3.org/ns/sosa/`
System Capabilities Module (of SSN)	`ssn-system`	`https://www.w3.org/ns/ssn/systems/`
Provenance Interchange Ontology	`prov`	`https://www.w3.org/ns/prov#`
Friend of a Friend Ontology	`foaf`	`http://xmlns.com/foaf/0.1/`
Dataset Usage Vocabulary	`duv`	`http://www.w3.org/ns/duv#`
Data Catalog Vocabulary	`dcat`	`https://www.w3.org/ns/dcat#`
Bibliographic Reference Ontology	`biro`	`http://purl.org/spar/biro/`
Web Annotation Ontology	`oa`	`http://www.w3.org/ns/oa#`

Table 1 shows excerpts of the XPaths realising the conceptual mapping. The test data[10] was created based on the example data for the SSN vocabulary[11], which was converted to RDF/XML using two online converters for two varying serialisations into RDF/XML. *MyBluemix RDF Validator and Converter*[12] uses `rdf:resource` attributes to define elements at one level (Listings 1), whereas *Easy RDF Converter*[13] uses the class names as XML elements and nests the objects (2). These examples illustrate the reason for the complexity of the XPaths, which allow both options to serialise triples from an RDF graph in RDF/XML.

In GCR, the API can be deployed in a container, which allows one to run the whole GEO label API with the existing Java Servlet[14]. In AWS Lambda, however, the Java Servlet application cannot be run, so a subset of the GEO label API was implemented with a bespoke request handling class[15]. This handler exposes the existing internal methods for generating SVGs based on URLs to metadata documents provided by the API caller. Then, the API, i.e., the request parameters and allowed HTTP methods, are configured in the Amazon API Gateway. Figure 1 shows a GEO label rendered by the prototype implementation developed as part of this work.

3.3 Performance Evaluation

Two usage scenarios were evaluated with an Apache JMeter[16] scripted test plan[17]. For all API queries, the URL of the example RDF serialisation file `MBC_all_factes_available_ ip68smartsensor.rdf` hosted on GitHub is passed via the `GET` request query parameter

---

[10] `https://github.com/nuest/GEO-label-java/tree/master/testdata`

[11] `https://www.w3.org/TR/vocab-ssn/#examples`

[12] `http://rdfvalidator.mybluemix.net/`

[13] `http://www.easyrdf.org/converter`

[14] `https://en.wikipedia.org/wiki/Java_servlet`

[15] See code module `lambda`: `https://github.com/nuest/GEO-label-java/tree/master/lambda`

[16] `http://jmeter.apache.org/`

[17] JMeter test plan file `GEO_Label_API.jmx` and the result files are available online at `https://github. com/nuest/GEO-label-java/tree/master/misc/JMeterTests`.

■ **Listing 1** Observation, converted with MyBluemix RDF Validator and Converter.

```
<rdf:Description
  rdf:about="http://example.org/data/iceCore/12#observation">

  <sosa:hasSimpleResult
    rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
      42
  </sosa:hasSimpleResult>
  <sosa:observedProperty
    rdf:resource="http://example.org/data/iceCore/12#CO2"/>
  <prov:wasAssociatedWith
    rdf:resource="http://example.org/data/Org/exampleOrg"/>
  <rdf:type
    rdf:resource="http://www.w3.org/ns/prov#Activity"/>
  <rdf:type
    rdf:resource="http://www.w3.org/ns/sosa/Observation"/>
</rdf:Description>

<rdf:Description
  rdf:about="http://example.org/data/Org/exampleOrg">

  <foaf:name>Example Organisation</foaf:name>
  <rdf:type rdf:resource="http://www.w3.org/ns/prov#Organization"/>
  <rdf:type rdf:resource="http://www.w3.org/ns/prov#Agent"/>
</rdf:Description>
```

`metadata` to the SVG-generating endpoint of the respective API deployments. The responses are deemed successful if the HTTP status code is 200 ("OK") and the content type is `image/svg+xml`. The test plan results for all conducted tests are published in the software repository[18]. An interactive app allows to inspect the plots of the results for all conducted tests, including ones not included in this article[19].

For both serverless computing providers, the default configurations were used for the evaluations. GCR allows users to configure the number of containers, the number of parallel requests handled by one container, and the required minimum response time. The GCR deployment used zone `europe-west1` with 256 Mebibyte working memory and 1 CPU, at a concurrency setting of 80. AWS Lambda starts more instances of a Lambda function as needed, limited by a configurable concurrency parameter (default value: 1000) for the number of running functions in the used region `eu-central-1`. The working memory on AWS is set to 1 Gibibyte with the default values for scaling[20].

Scenario A simulates a geospatial catalogue service with 1000 users whose browsing of the catalogue user interface results in 1 request per second per user. Figures 2 and 3 show the response times during the test execution for GCR and AWS Lambda, respectively[21]. All sent requests have a non-failure status code (`HTTP 200`). The two different colours in the plots denote the requests that take less than ("Success") or longer than ("Failure") 1 second.

---

[18] JMeter test plan file `GEO_Label_API.jmx` and the result files are available online at `https://github.com/nuest/GEO-label-java/tree/master/misc/JMeterTests`.
[19] `https://geolabel-ssno-paper.herokuapp.com/`
[20] `https://docs.aws.amazon.com/lambda/latest/dg/scaling.html`
[21] Data loading and plot functions are based on code from the R package `loadtest` [21].

■ **Figure 1** GEO label based on SSW sensor metadata, rendered by the GCR deployment, with all eight facets fulfilled; source URL: https://glbservice-nvrpuhxwyq-ew.a.run.app/glbservice/api/v1/svg?metadata=https://raw.githubusercontent.com/nuest/GEO-label-java/master/server/src/test/resources/ssno/ERC_all_factes_available_ip68smartsensor.rdf.
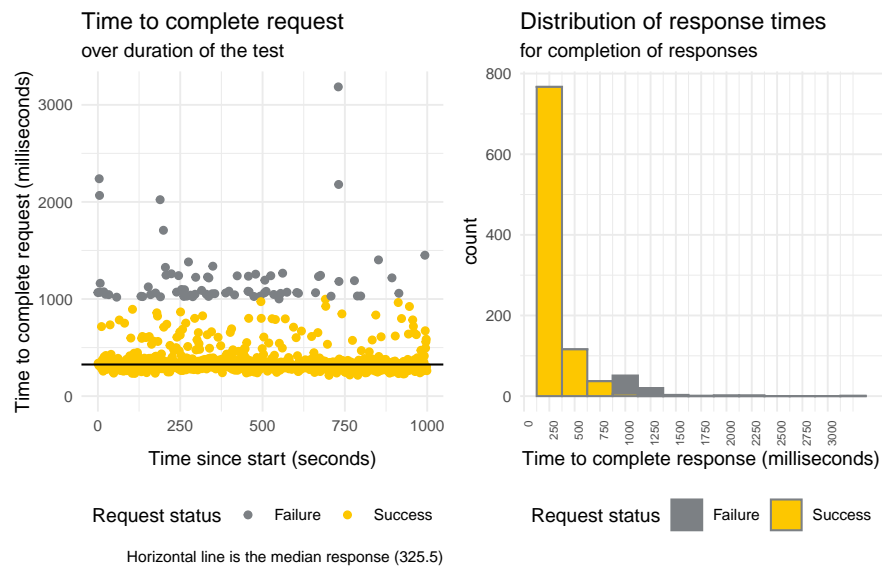
■ **Figure 2** Plots of response time for GCR deployment under scenario A: elapsed time to complete requests (left); histrogram with distribution of response times (right); result data file: `GCR_Scenario_2_V1`.

■ **Listing 2** Observation, converted with Easy RDF Converter.

```
<sosa:Observation
  rdf:about="http://example.org/data/iceCore/12#observation">

  <rdf:type rdf:resource="http://www.w3.org/ns/prov#Activity"/>
  <prov:wasAssociatedWith>
    <prov:Agent
      rdf:about="http://example.org/data/Org/exampleOrg">

      <rdf:type
        rdf:resource="http://www.w3.org/ns/prov#Organization"/>
      <foaf:name>Example Organization</foaf:name>
    </prov:Agent>
  </prov:wasAssociatedWith>
  <sosa:observedProperty>
    <rdf:Description
      rdf:about="http://example.org/data/iceCore/12#CO2">
      <ssn:isPropertyOf
        rdf:resource="http://example.org/data/iceCore/12"/>
    </rdf:Description>
  </sosa:observedProperty>

  <sosa:hasSimpleResult
    rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
      42

</sosa:Observation>
```

This threshold is used because interactions below one second were found to not interrupt a user's train of thought and are therefore suitable for interactive use [20]. The mean times to complete the request are 414 seconds for GCR and 943 seconds for AWS Lambda.

Scenario B tests the batch generation of labels where an operator of a sensor catalogue wants to generate 100 labels at once. Here we measure the overall time for processing all requests, and the operations were repeated 5 times. There is no threshold as in Scenario A. For GCR, this led to failures due to the memory limit; but, the test was completed with a memory of 1 Gibibyte and 2 CPUs per container instance. The resulting data is shown in Figure 4. GCR's need for additional resources can be traced back to an overhead of the full Java Servlet, which the Lambda function handler, which is comparably more minimal, does not suffer from. With the increased resources in GCR, the duration was up to 45 seconds for the first run and decreased though to only about 3 seconds for the fifth repetition. For AWS Lambda, the processing took about 8 seconds on the first run and dropped to around 1 second in the second to fifth repetitions, as shown in Figure 5.

A variant of the batch generation is a test scenario with 1000 parallel requests. This scenario could not be completed by either platform with the maximum available hardware configurations. The error messages (`Connection reset` and `SSL handshake terminated`) hint that the services blocked the large number of parallel requests, such that users would need more powerful (and more costly) deployments. Reducing the number of parallel requests eventually led to successful scenario executions at 600 requests in 51 seconds for GCR and 300 requests in 9 seconds for AWS Lambda (see data files `GCR_Scenario_4_2_V3` and `AWS_Scenario_4_2_V4`).

10   Serverless GEO labels for the SSW
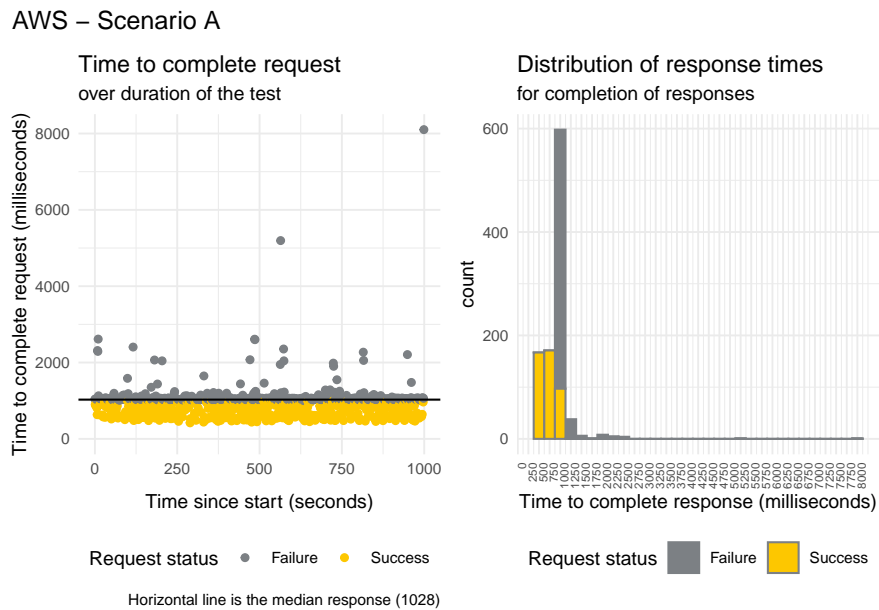
Figure 3 Plots of response time for AWS deployment in scenario A: elapsed time to complete requests (left); histrogram with distribution of response times (right); result data file: `AWS_Scenario_2_V1`.
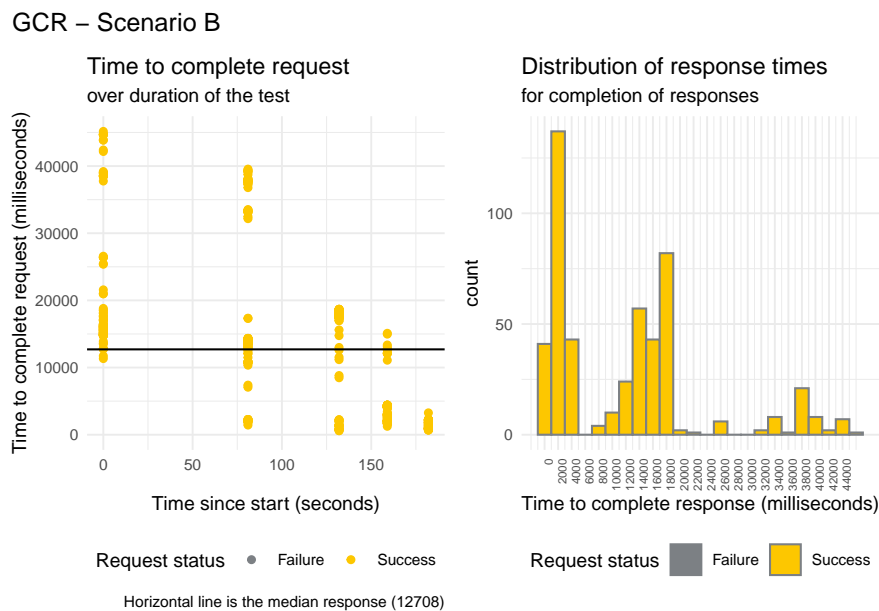
Figure 4 Plots of response time for GCR deployment in scenario B: elapsed time to complete requests (left); histrogram with distribution of response times (right; please note the different bin size compared to other plots); result data file: `GCR_Scenario_3_V2`.

AWS – Scenario B

Time to complete request
over duration of the test

Distribution of response times
for completion of responses
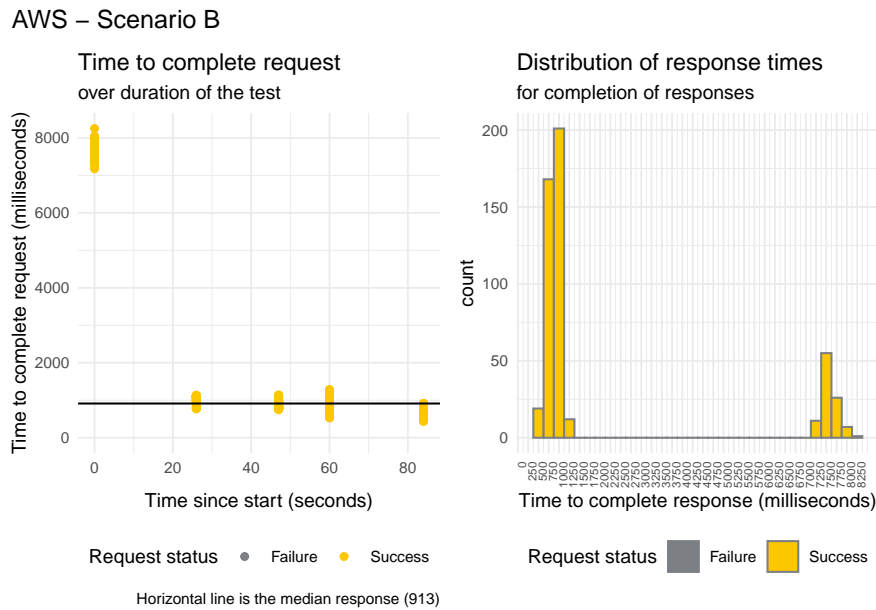

Figure 5 Plots of response time for AWS deployment in scenario A: elapsed time to complete requests (left); histrogram with distribution of response times (right); result data file: `AWS_Scenario_3_V2`.

4   Discussion

The **mapping** of GEO label facets to properties in the Semantic Sensor Web was an iterative process. While we were able to find data sources for all GEO label facets, the mapping is limited by the availability of realistic SSW datasets. First, the variability of real-world data may not be adequately captured. Second, and the nature of the mapping does not capture cases where concepts between the GEO label's facets do not unambiguously match concepts behind SSW elements. Compared to the centrally managed data sources and industry-driven OGC standards of the original GEO label, we find no need to make distinctions between metadata given by providers and by third-parties, e.g., commenting servers. However, such multi-stakeholder perspectives could mitigate shortcomings in the creation process of the GEO label mapping for SSW. More real-world metadata could improve the scope of the facet data sources, e.g., by deriving from common practices if a comment is actually about a relevant part of a sensor's properties, and not about some less relevant part of the RDF graph. The taken iterative, example-based approach could also be contrasted with the initial creation of an ontology for the GEO label facets and then aligning the GEO label ontology with existing (SSW) ontologies. The alignment-based approach could also improve the scalability of the mapping for a larger variety of uses cases and SSW datasets.

Concerning the mapping's **implementation**, we found that a document-based approach using RDF/XML could be built quickly on the existing implementation. However, such an approach does not leverage the power of the Semantic Web, e.g., reasoning on dynamically built graphs and aligned ontologies. Furthermore, for some facets, e.g., producer profile, there were clear complications of relying on serialisations into an RDF/XML document. A different approach based on native Semantic Web technologies, such as SPARQL [13], could help address the limited coverage of the presented mapping, and in addition take

into account relationships between linked resources by, for example, measuring the distance between connected resources in a graph. The GEO label's option to have "half-filled" facets, which denotes availability of information at a higher level, could expose such more complex scenarios. Most critically, the presented approach is limited by the design process starting only from the current GEO label facets. That is why specific discovery challenges of the SSW may not be adequately addressed. While the label itself may be interactive, the majority of information behind the label is seen as rather static. This may partly be attributed to the GEO label's origin in GEO, with more traditional roles of provider and user. The SSW's potentially very dynamic nature, for examples live data streams, and flexible distributed architecture, in which anybody can create and publish new ontologies and datasets, may require additional facets or a more sophisticated presentation of sources and currentness of the data behind a label.

The **evaluation** results of Scenario A show show no discernible cold start effect, as one might have expected, where resources need to be activated for the first request or additional resources are added over time. Only few requests take over 1 second to complete and only relatively few outliers exists on the same order of magnitude. For AWS Lambda, both mean and median of elapsed time to complete a request are close to 1 second. For GCR, the elapsed time is well below 0.5 seconds. These results imply that the serverless label generation is suitable for interactive use, with slight advantages of GCR which has overall shorter durations. A limitation for this scenario is that only the generation of the label is tested, whereas for users additional time would be taken up by the client-side rendering of the images. The effects of dropping durations for batch processing in Scenario B were likely achieved by a combination of autoscaling in the underlying platforms and the built-in caching of the GEO label API Java Servlet. Especially on AWS Lambda, the drop after the first iteration is considerable, even tough no internal caching mechanism exists.

Regarding the platforms we used, the data might further point to an advantage for the reduced implementation of the AWS Lambda functions compared to the full Java Servlet running in containers on GCR, though both showed scaling mechanisms of serverless computing to be effective. The results make clear that specific evaluations for each use case and platform are warranted. More test scenarios could include varying allocated resources at the cloud providers to optimise performance versus costs, touching both on the respective configuration parameters and the client-side implementation. For example, the steep drop in Scenario B on AWS Lambda could be used to warm up a service instance, which may have relatively small resources, with a portion of the data for batch processing, and then following up with several chunks afterwards. In contrast, the shorter average response times of GCR may make it more suitable for a scenario with more constant load even if fewer resources are allocated.

5   Conclusions

In this study, we transferred the goal of the GEO label, which is to improve data discovery by providing a visual overview of available information in machine-readable metadata, to the Semantic Sensor Web. While we were able to find data sources for all GEO label facets using a document-centric approach, the mapping is limited by available datasets and does not leverage the potential of using reasoning in the SSW. Ideally, the creation of a more sustainable mapping and potentially even adaptation of GEO label facets in the future is based on a larger body of public sensor metadata in SSNO format, on a consultation of multiple stakeholders, and on a complementary perspective derived from the SSW's discovery challenges.

We found that the serverless platforms proved suitable for realistic test scenarios, though, naturally, the used free tiers have limits. It became also clear that the different cost models and configurations make serverless solutions difficult to compare. Future evaluations may utilise a strictly cost-based comparison of scenarios with resources tuned to deliver similar performance in the user-facing API.

Finally, the usefulness of the GEO label remains to be demonstrated in broad deployments with many users and extensive user studies. With the practical solutions for label generation introduced in this work, the actual spreading of labels will require leading organisations to add and maintain labels on their widely used geospatial catalogues. In the meantime, a bottom-up approach with client-side label integration [23] could provide the benefits of GEO labels to interested users, and the GEO label can be examined in relation to recent developments on scientific data publication such as the FAIR Guiding Principles [26].

References

1   Grigori Babitski, Simon Bergweiler, Jörg Hoffmann, Daniel Schön, Christoph Stasch, and Alexander C. Walkowski. Ontology-Based Integration of Sensor Web Services in Disaster Management. In Krzysztof Janowicz, Martin Raubal, and Sergei Levashkin, editors, *GeoSpatial Semantics*, Lecture Notes in Computer Science, pages 103–121, Berlin, Heidelberg, 2009. Springer. `doi:10.1007/978-3-642-10436-7_7`.

2   Ioana Baldini, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche Ishakian, Nick Mitchell, Vinod Muthusamy, Rodric Rabbah, Aleksander Slominski, and Philippe Suter. Serverless Computing: Current Trends and Open Problems. In Sanjay Chaudhary, Gaurav Somani, and Rajkumar Buyya, editors, *Research Advances in Cloud Computing*, pages 1–20. Springer, Singapore, 2017. `doi:10.1007/978-981-10-5026-8_1`.

3   Mike Botts, George Percivall, Carl Reed, and John Davidson. OGC® sensor web enablement: Overview and high level architecture. In *GeoSensor Networks*, pages 175–190. Springer Berlin Heidelberg, 2008. `doi:10.1007/978-3-540-79996-2_10`.

4   Dan Brickley and Libby Miller. FOAF Vocabulary Specification. Technical report, January 2014. URL: `http://xmlns.com/foaf/spec/`.

5   Jean-Paul Calbimonte, Hoyoung Jeung, Oscar Corcho, and Karl Aberer. Semantic sensor data search in a large-scale federated sensor network. In *Proceedings of the 4th international workshop on semantic sensor networks*, volume 839 of *CEUR Workshop Proceedings*, pages 23–38, Bonn, Germany, 2011. URL: `http://ceur-ws.org/Vol-839/calbimonte.pdf`.

6   Eliot J. Christian. GEOSS Architecture Principles and the GEOSS Clearinghouse. *IEEE Systems Journal*, 2(3):333–337, September 2008. Conference Name: IEEE Systems Journal. `doi:10.1109/JSYST.2008.925977`.

7   J. Clark and S. DeRose. XML Path Language (XPath), Version 1.0. W3C Recommendation, November 1999. URL: `http://www.w3.org/TR/xpath`.

8   Erik Dahlström, Patrick Dengler, Anthony Grasso, Chris Lilley, Cameron McCormack, Doug Schepers, and Jonathan Watt. Scalable Vector Graphics (SVG) 1.1 (Second Edition). W3C Recommendation, 2011. URL: `http://www.w3.org/TR/SVG/`.

9   Angelo Di Iorio, Andrea Nuzzolese, Silvio Peroni, David Shotton, and Fabio Vitali. Describing bibliographic references in RDF. In Alexander Garc{\'i}a Castro, Christoph Lange, Phillip Lord, and Robert Stevens, editors, *4th Workshop on Semantic Publishing (SePublica 2014)*, volume 1155 of *CEUR Workshop Proceedings*, Anissaras, Greece, May 2014. URL: `http://ceur-ws.org/Vol-1155/paper-05.pdf`.

10 Bernadette Farias Lóscio, Eric G. Stephan, and Sumit Purohit. Data on the Web Best Practices: Dataset Usage Vocabulary. Technical report, 2016. Library Catalog: www.w3.org. URL: `https://www.w3.org/TR/vocab-duv/`.

11 Anika Graupner. Ein Metadatenlabel für das semantische Sensorweb. April 2020. `doi:10.31237/osf.io/fs48a`.

12 Armin Haller, Krzysztof Janowicz, Simon J. D. Cox, Maxime Lefrançois, Kerry Taylor, Danh Le Phuoc, Joshua Lieberman, Raúl García-Castro, Rob Atkinson, and Claus Stadler. The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. *Semantic Web*, 10(1):9–32, January 2019. `doi:10.3233/SW-180320`.

13 Steve Harris and Andy Seaborne. SPARQL 1.1 Query Language. W3C Recommendation, March 2013. URL: `https://www.w3.org/TR/sparql11-query/`.

14 Krzysztof Janowicz, Sven Schade, Arne Bröring, Carsten KeSSler, Patrick Maué, and Christoph Stasch. Semantic Enablement for Spatial Data Infrastructures. *Transactions in GIS*, 14(2):111–129, 2010. `doi:10.1111/j.1467-9671.2010.01186.x`.

15 Hoyoung Jeung, Sofiane Sarni, Ioannis Paparrizos, Saket Sathe, Karl Aberer, Nicholas Dawes, Thanasis G. Papaioannou, and Michael Lehning. Effective Metadata Management in Federated Sensor Networks. In *2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pages 107–114, June 2010. `doi:10.1109/SUTC.2010.29`.

16 Simon Jirka, Arne Bröring, and Christoph Stasch. Discovery Mechanisms for the Sensor Web. *Sensors*, 9(4):2661–2681, April 2009. `doi:10.3390/s90402661`.

17 Timothy Lebo, Satya Sahoo, and Deborah McGuinness. PROV-O: The PROV Ontology. Technical report, April 2013. Publisher: W3C. URL: `https://www.w3.org/TR/prov-o/`.

18 Victoria Lush. *Visualisation of quality information for geospatial and remote sensing data: providing the GIS community with the decision support tools for geospatial dataset quality evaluation.* PhD thesis, Aston University, 2015. URL: `https://research.aston.ac.uk/en/studentTheses/visualisation-of-quality-information-for-geospatial-and-remote-se`.

19 Victoria Lush, Lucy Bastin, and Jo Lumsden. Developing a geo label: providing the gis community with quality metadata visualisation tools. *Proceedings of the 21st GIS Research UK (GISRUK 3013), Liverpool, UK*, pages 3–5, 2013. URL: `https://www.geos.ed.ac.uk/~gisteac/proceedingsonline/GISRUK2013/gisruk2013_submission_44.pdf`.

20 Jakob Nielsen. Response Times: The 3 Important Limits, January 1993. URL: `https://www.nngroup.com/articles/response-times-3-important-limits/`.

21 Jacqueline Nolis. *loadtest: HTTP load testing directly from R*, 2020. R package version 0.1.2. URL: `https://github.com/tmobile/loadtest`.

22 Daniel Nüst and Anika Graupner. nuest/GEO-label-java: Release 0.3.0, February 2020. `doi:10.5281/zenodo.3673870`.

23 Daniel Nüst, Lukas Lohoff, Lasse Einfeldt, Nimrod Gavish, Marlena Götza, Shahzeib Tariq Jaswal, Salman Khalid, Laura Meierkort, Matthias Mohr, Clara Rendel, and Antonia van Eek. Guerrilla Badges for Reproducible Geospatial Data Science. AGILE Short Papers, Limassol, Cyprus, 2019. AGILE. `doi:10.31223/osf.io/xtsqh`.

24 Daniel Nüst and Victoria Lush. A GEO label for the Sensor Web. AGILE Short Papers, Lisbon, Portugal, 2015. AGILE. `doi:10.31223/osf.io/ka38z`.

25 Amit Sheth, Cory Henson, and Satya S. Sahoo. Semantic Sensor Web. *IEEE Internet Computing*, 12(4):78–83, July 2008. `doi:10.1109/MIC.2008.87`.

26 Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J. G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A. C. t Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018, March 2016. `doi:10.1038/sdata.2016.18`.