Monitoring the Status of SPARQL Endpoints

Pierre-Yves Vandenbussche[1], Carlos Buil Aranda[2],
Aidan Hogan[3], and Jürgen Umbrich[1]

[1] Fujitsu (Ireland) Limited, Swords, Co. Dublin, Ireland
[2] Department of Computer Science, Pontificia Universidad Católica de Chile
[3] Digital Enterprise Research Institute, National University of Ireland, Galway

Abstract. We demo an online system that tracks the availability of over four-hundred public SPARQL endpoints and makes up-to-date results available to the public. Our demo currently focuses on how often an endpoint is online/offline, but we plan to extend the system to collect metrics about available meta-data descriptions, SPARQL features supported, and performance for generic queries.

1 Motivation

In previous work [2], we presented an analysis of the landscape of public SPARQL endpoints and asked the question: are these endpoints ready for action?[4] Taking the full list of 427 public endpoints from the CKAN/DataHub catalogue (as available at the time of writing), for each endpoints, we conducted a number of experiments to gauge the following four main aspects:

Discoverability: What kinds of meta-data descriptions are available about the endpoints and their content? How easy are these descriptions to find?
Interoperability: Which SPARQL (1.1) features does each endpoint support? Which features (or combinations of features) lead to exceptions?
Efficiency: How do the endpoints perform for answering generic forms of queries? How is cold-cache performance vs. warm-cache performance? What is the latency like over HTTP?
Availability: What are the average uptimes of the endpoints? How many endpoints are dying/have died? How many endpoints have high reliability?

Our results showed that about half of the endpoints listed on CKAN/-DataHub are now offline, that only a few endpoints make meta-data descriptions available about their content (VoID) or features supported (SPARQL 1.1 Service Descriptions) in easy-to-find locations, that there was mixed adoption

[4] This work is accepted for the Experiments track of ISWC 2013 [2]. This demo paper rather focuses on our tool for making results available to the community.

of SPARQL and (recently standardised) SPARQL 1.1 features, that the performance of different endpoints over HTTP for generic queries could vary by orders of magnitude, and that less than one third of the endpoints had an average availability in the interval 99–100% (i.e., at least two-nines availability). We concluded that the usability of different public endpoints varies greatly.

We thus propose a system that tracks and collects metrics about public endpoints over time. Currently, our service tracks the hourly availability of endpoints, and we plan to extend it to collect weekly metrics about the available meta-data, supported features and performance of these endpoints, as well as other metrics that the community may wish to suggest.

In Section 2, we first discuss our current "SPARQL Endpoint Status" system, available online at `http://labs.mondeca.com/sparqlEndpointsStatus/`. Thereafter, in Section 3, we discuss our proposed extensions.

2   SPARQL Endpoint Status

Monitoring Availability The system automatically collects and updates a list of public SPARQL endpoints from the CKAN/DataHub catalogue. These endpoints are queried on an hourly basis using two alternative SPARQL queries:

`ASK WHERE{ ?s ?p ?o . }`	`SELECT ?s WHERE{ ?s ?p ?o . } LIMIT 1`

The `ASK` query on the left is issued first. If this query fails (from previous experience, we note that some endpoints do not support `ASK` [2, § 3]), we try the `SELECT` query on the right. Both queries are selected at they should be as cheap as possible for the endpoint to run: our goal is simply to check whether or not the endpoint is available for answering queries. If the endpoint returns a valid SPARQL response for either query, we then say that the endpoint is available at that timepoint. We also record the time taken for the query to execute.

At the time of writing, we have collected more than two million hourly pings across hundreds of endpoints over a period of more than two years. Detailed analysis of these availability results is available in [2, § 5].

User Interface We provide a user interface to browse and visualise the hourly results. The user interface supports two primary views.

The first view, exemplified in Figure 1, provides a full list of all the monitored endpoints, their availability in the past 24 hours (ratio of successful hourly queries in that period), and their availability in the past seven days. A green/yellow/red/gray icon indicates, resp., that the endpoint is operating normally/available but had problems in the past 24 hours/not available currently/not available once in the past 24 hours. As per the icons listed on the right of the screenshot, each endpoint is also associated with (1) an RSS feed to provide updates on availability information, (2) a link to the endpoint itself and (3) a link to the relevant CKAN/DataHub page for the dataset it relates to.

The second view provides details for a given endpoint. Figure 2 shows an example screenshot for the DBpedia endpoint. The graph on the left shows the
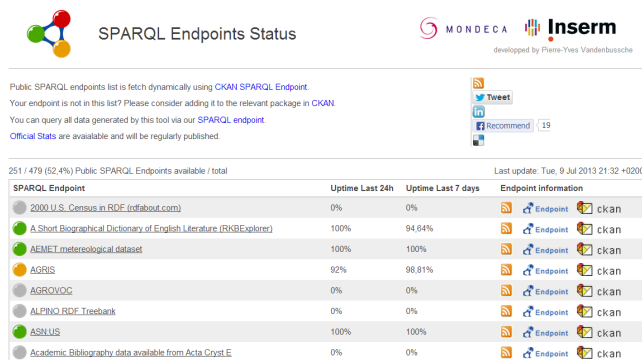
Fig. 1. Screenshot of current SPARQL Endpoint Status list

response times for the last 24 hourly pings to that endpoint. The graph on the right plots the 24 hour availability for each of the last seven days.
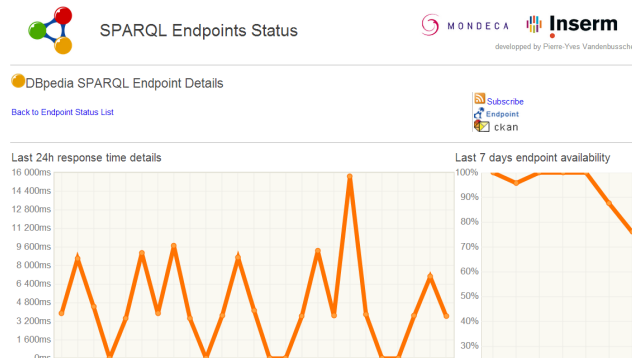

Fig. 2. Screenshot of current SPARQL Endpoint Status detail view for DBpedia

RDF Meta-data Results of the hourly pings are exported as RDF. Figure 3 presents an example description. We reuse existing vocabularies as much as possible (VoID, dcterms, etc.) to describe each dataset, their related SPARQL endpoint, title and identifier, etc. To capture availability information, we designed a new vocabulary (no existing one handled this feature). The "endpoint status" vocabulary[5] (*ends*) allows the description of a status observation with the information of date, description (we are here reusing dcterms vocabulary), status availability and response time. All RDF data are then published in a SPARQL Endpoint available at: http://labs.mondeca.com/endpoint/ends.
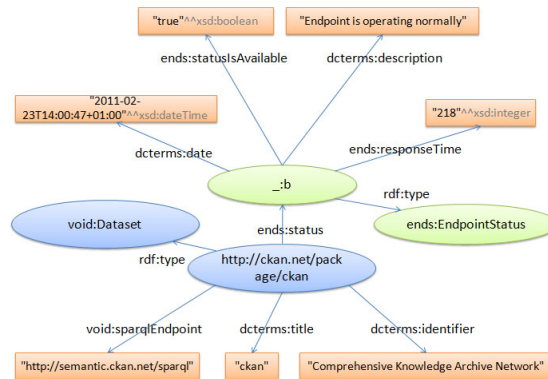
---

[5] http://labs.mondeca.com/vocab/endpointStatus/

Fig. 3. Schema used to express SPARQL endpoint availability in RDF

3 Future Extensions

Our system currently captures endpoint availability and query latency. In line with the discussion of Section 1 and the methods of our experimental paper [2], we wish to extend our system to track more metrics about public endpoints. These would include: (1) what meta-data descriptions about each endpoint/dataset are available and where (e.g., VoID, SPARQL 1.1 SD), (2) what query features each endpoint supports (e.g., SPARQL 1.1, full-text), (3) what performance can be expected for generic queries (atomic lookups, dump queries, controlled joins). Since the queries are more expensive to run, we propose running them on a weekly basis to not overburden endpoints. We would then extend our UI and RDF vocabulary to make these metrics available. We are very much open to suggestions/use-cases from the community for collecting further metrics. Furthermore, we are considering making a locally deployable version for clients to monitor endpoints of relevance to them.

References

1. K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao. Describing linked datasets. In *LDOW*, 2009.
2. C. B. Aranda, A. Hogan, J. Umbrich, and P.-Y. Vandenbussche. SPARQL Web-Querying Infrastructure: Ready for Action? In *ISWC*. Springer (LNCS), 2013. (Accepted; to appear.).
3. G. T. Williams. SPARQL 1.1 Service Description. W3C Recommendation, March 2013.