BARTOC FAST
A Federated Asynchronous Search Tool for Remote Vocabulary Access

Abstract:
In this paper we introduce BARTOC FAST, a federated asynchronous search tool for remote vocabulary access. We first motivate the need for BARTOC FAST by exposing the limitations of the local BARTOC.org Skosmos instance. We then discuss the advantages of BARTOC FAST – vast search space, low footprint, and modularity – and provide an overview over its implementation and some design challenges. We close by considering some anticipated use cases and plans for future development.

1.0 Limitations of BARTOC.org
The Basic Register of Thesauri, Ontologies & Classifications (BARTOC.org) is a full terminology registry for knowledge organization systems (KOS). We aim to collect as many controlled vocabularies as possible in one place, to describe them uniformly and to make them accessible. So far this is not the case, at least not with regard to the searchability of vocabulary content. BARTOC FAST attempts to supplement BARTOC.org with an application that allows to search for concepts and terms in vocabularies. The whole project stems from our conviction that KOS presented in BARTOC.org should comply with the FAIR principles for the use of controlled vocabularies.

BARTOC.org consists of two main modules: one contains the metadata of currently about 3,200 KOS, the other provides the members of the KOS, i.e. concepts and terms, as long as they are available in SKOS format. For the SKOS vocabulary service we use Skosmos, an open source web-based SKOS publishing tool developed by Suominen et al. (2015).

Skosmos allows users to browse SKOS vocabularies alphabetically or hierarchically and offers structured concept display. For BARTOC.org, the most important feature was the global search, which enabled searching for concepts across all hosted vocabularies. Consequently, vocabularies from other terminology registries had to be uploaded to our own RDF Triple Store. This caused serious problems after some time:

- Skosmos could no longer process the large number of 1,436 vocabularies (3,314,740 concepts, 11,492,219 terms, with the massive Getty vocabularies not included).
- Since our vocabularies were mostly clones of remotely hosted vocabularies, it was a considerable effort to keep them up to date by periodically checking for updates or new versions. Comparable portals such as Linked Open Vocabularies face similar problems and solve them through manual annual reviews and comments in a separate metadata field.

References:
1. https://bartoc.org
3. https://lov.linkeddata.es/dataset/lov/vocab/airs
• Not all vocabularies were available in full SKOS. We had to realize that Skosmos is a great tool, but was no longer sufficient for our special purpose of a powerful search instrument for plenty of KOS. Or as Osma Suominen puts it in the Skosmos User Forum: “Skosmos works well up to around tens of thousands of concepts (...). With more than 100,000 concepts, it starts getting slow” (Suominen 2015).

One could add that with millions of concepts the global search function collapses. However, smaller Skosmos deployments containing unique vocabularies, run by institutions like UNESCO⁴, Food and Agriculture Organization (FAO) of the United Nations⁵, National Library of Finland⁶, University of Oslo Library⁷, Inist-CNRS⁸, etc. provide REST-style APIs and Linked Data access to the underlying data. Our task was to replace the global search of Skosmos with a federated search method that was capable of querying multiple REST APIs and SPARQL endpoints simultaneously to make millions and millions of concepts from any number of terminology registries accessible with one tool.

2.0 BARTOC FAST

We have overcome the aforementioned limitations of BAROC.org by developing BARTOC FAST⁹. BARTOC FAST is a remote retrieval aid for thesaurus, ontology and classification concepts. The acronym FAST stands for Federated Asynchronous Search Tool and should not be confused with OCLC’s vocabulary of the same name, which resolves to Faceted Application of Subject Terminology¹⁰. At present the FAST federation contains more than 20 resources – including BARTOC.org Skosmos¹¹, Getty Vocabularies¹², the Integrated Authority File (GND) via lobid-gnd, as introduced by Steeg et al. (2019), and the Research Vocabularies Australia¹³, see https://bartoc-fast.ub.unibas.ch/bartocfast/about for a full list – thus comprising a vast search space.

BARTOC FAST offers three decisive advantages over BARTOC.org. First, Skosmos can take a back seat and exclusively serve as an instance for SKOS vocabularies that are not hosted anywhere else. Secondly, the data in BARTOC FAST is always up-to-date, as it comes directly from the APIs of the terminology registries. Conversely the footprint of BARTOC FAST as compared to BARTOC.org is massively reduced since the maintenance and support of the terminology registries is delegated to their providers. Thirdly, BARTOC FAST is a single access point that allows users to search not only for vocabularies but also in vocabularies. The new search interface of BARTOC.org will combine both functionalities (see Figure 1).

---

⁴ http://vocabularies.unesco.org/browser/
⁵ http://agrovoc.uniroma2.it/agrovoc/
⁶ https://finto.fi/
⁷ http://data.ub.uio.no/skosmos/nb/
⁸ https://www.loterre.fr/skosmos/
⁹ https://bartoc-fast.ub.unibas.ch/bartocfast/
¹⁰ https://www.oclc.org/research/themes/data-science/fast.html
¹¹ https://bartoc-skosmos.unibas.ch/
¹² https://www.getty.edu/research/tools/vocabularies/
¹³ https://vocabs.ands.org.au/
2.1 Implementation

BARTOC FAST is implemented in Python 3.7 and runs on the servers of the Basel University Library. The frontend and backend will be discussed in turn.

The BARTOC FAST frontend uses the Django framework\textsuperscript{14} and comes with three distinct views: basic, advanced and API. Each view corresponds the expectations of a specific type of user:

1. The basic view mirrors the user interface of a discovery tool such as Primo\textsuperscript{15}. All parameters (except the search input of course) are fixed and set to values that yield usable results in most cases. The user simply types in a search word and receives a list of results. This list is both sortable and searchable (see Figure 2) and hence allows for additional refinement.

2. The advanced view is similar to the basic view but allows for a customization of all parameters (see Figure 3). These include the maximum search time, the choice of queried resources, and the option of keeping duplicates. Note that some desirable facets, filters and modifiers are not yet implemented but scheduled for the next development cycle.

The BARTOC FAST API is a HTTP-based RESTful API, soon to be compliant with the Reconciliation Service API\textsuperscript{16} returning results as JSKOS, a data format for KOS by Voß (2019), or generic JSON-LD. The view itself is identical to the advanced view.

\textsuperscript{14} https://www.djangoproject.com/
\textsuperscript{15} https://www.exlibrisgroup.com/products/primo-discovery-service/
\textsuperscript{16} https://github.com/OpenRefine/OpenRefine/wiki/Reconciliation-Service-API/
Figure 2: BARTOC FAST basic view results list for search term “knowledge organization”.

Shifting the focus towards the BARTOC FAST backend now, since BARTOC FAST is a remote service, it needs to process many distinct APIs. For this reason, resource modelling and query resolving in BARTOC FAST is handled by a GraphQL\(^\text{17}\) schema which makes use of the Graphene module\(^\text{18}\). Simply put, GraphQL provides a (meta-)query language over all the resources in the BARTOC FAST federation (see Figure 4). Resource modelling is modular and on an API basis. This means that new instances of already modelled APIs (such as Skosmos and SPARQL) can easily be added to the BARTOC FAST federation.

\(^{17}\) https://graphql.github.io/graphql-spec/June2018/
\(^{18}\) https://graphene-python.org/
BARTOC FAST resolves federated queries by translating the user’s search input into an API call for each resource; the list of aggregated results is then returned. More precisely, query resolution takes three steps for each resource: asynchronously fetching data by means of an API call to the resource, normalizing this data, and purging duplicates. Every result in the list of aggregated results has three mandatory data fields: a URI, a nonempty set of labels, and a source:

1. The URI is used to tell apart results within and across resources. Given the results of a single resource, all results with the same URI are merged into a single result and their labels (respectively the contents of their labels) are aggregated. For the results of multiple resources, redundant results as identified per URI are purged.

2. BARTOC FAST results use four SKOS labels\(^\text{19}\) as base, namely skos:prefLabel, skos:altLabel, skos:hiddenLabel and skos:definition. Since not all resources in BARTOC FAST employ SKOS, the semantic equivalents of these

\(^{19}\) https://www.w3.org/2009/08/skos-reference/skos.html
labels are provided via a mapping (e.g., skos:prefLabel is equivalent to rdfs:label\textsuperscript{20} is equivalent to gnd:preferredName\textsuperscript{21}). Note that the descriptor takes no special priority.

3. The source is the queried resource.

2.2 Challenges

In this section we discuss two design challenges that we encountered in the development of BARTOC FAST.

An initial challenge concerns the construction of federated queries. Generally speaking, a federated query processes the input and passes it to each resource in the federation in a format that is accepted by the resource. So when a BARTOC FAST search request is carried out, the user’s search word is transformed into an API call for each resource in the BARTOC FAST federation. Since each API is represented as a model, this task is reduced to transforming the input string into a search request for each model. However, different models allow for different kinds of search requests. To give two examples: not all models support Boolean operators, and different labels are given varying importance in different models. Our solution to this challenge is twofold. First, instead of trying to construct formally equivalent search requests across all models, we opted for the pragmatic approach of constructing search requests with similar outputs and behaviors. Second, users can toggle a view of the exact API calls triggered by their string input on the BARTOC FAST results page. This whiteboxing awards users more control since unwanted API calls can simply be turned off. In the future we plan to include options for customizing federated queries (e.g., exact match over label X).

A second challenge concerns the varying sizes of the resources within the BARTOC FAST federation. Unsurprisingly, big registries such as the Integrated Authority File (GND) tend to outcrowd smaller registries with respect to the number of results for a search request. However, the number of results is not an indicator of quality. Generally speaking, there are at least two solutions available to this problem:

1. Expand or shrink the BARTOC FAST federation according to need. We have already implemented this solution by allowing the user to manually (de)select queried resources in the advanced view as discussed above.

2. Rank the results. Some simple variants for ranking include giving preference to label X over label Y, or giving preference to results with less empty labels. In addition to reducing noise, this solution has an additional advantage: if a resource is still dominant for ranked results, it is a marker of the resource’s quality rather than a problem. The downside of this solution is that the neutrality of the results is no longer guaranteed. The role of BARTOC FAST would shift from an aggregator to a curator. For this reason, if we decide to implement this solution, it will be strictly opt-in.

3.0 Outlook

At its current stage, BARTOC FAST is still a prototype with limited service, like a Google search kind of tool for KOS content. In this preliminary stage of development,

\textsuperscript{20} http://www.w3.org/2000/01/rdf-schema#label

\textsuperscript{21} https://d-nb.info/standards/elementset/gnd#preferredName
it does not fully benefit from all SKOS labels with the REST API queries or RDF data with the SPARQL queries, which allow different query types, plus the filters, aggregates, modifiers, and operators. Such extended functionalities will be part of future development steps.

However, with BARTOC FAST in production, we pay particular attention to possible use cases now. BARTOC FAST provides a valuable vocabulary resource for KOS mapping (e.g. via Cocoda mapping tool by Balakrishnan et al. (2018)) and automated subject indexing (e.g. via Annif by Suominen (2019)) of multidisciplinary digital repositories like Zenodo22 and similar, that don’t come with a controlled vocabulary natively. We also intend to further grow the BARTOC FAST federation by adding instances of already modelled APIs or by modelling APIs that promise to add value in scope or depth. If you know of a resource that should be added to BARTOC FAST, please do not hesitate to contact us! Finally, we plan to provide full access to the BARTOC FAST source code under a permissive license as soon as some security issues have been addressed.

References

---

22 https://zenodo.org/