Harmonizing services for LOD vocabularies: a case study

Ghislain Auguste Atemezing[1], Bernard Vatant[2], Raphaël Troncy[1] and Pierre-Yves Vanderbussche[3]

[1] EURECOM, Sophia Antipolis, France, <firstName.lastName@eurecom.fr>
[2] Mondeca, Paris, France, <bernard.vatant@mondeca.com>
[3] Fujitsu, Galway, Ireland, <py.vanderbussche@fujitsu.com>

Abstract. Vocabularies are more and more (re)-used in the Linked Data ecosystem. However, managing the prefixes associated to their Uniform Resource Identifiers (URIs) is still cumbersome and namespaces are sometimes referring to different pair <prefix, URI>. In this paper, we propose to align two well-known services with the aim of managing and harmonizing vocabularies' namespaces. We use prefix.cc that provides a look up service for namespaces in general and Linked Open Vocabularies (LOV) that extracts vocabularies metadata. Our method enables to identify three different scenarios: (i) conflicts between prefix.cc and LOV; (ii) prefixes in LOV not present in prefix.cc and (iii) URIs in prefix.cc that are actually LOV-able vocabularies. We describe how we solve each of these issues, with actions ranging from updating the different services to contacting the editors of the vocabularies to fix clashes among prefixes. Finally, we present the new LOV API that enables to check whether those namespaces in prefix.cc can actually be vocabularies to be inserted in the LOV ecosystem or not.

Keywords: Vocabulary discovery, Linked Open Vocabularies, prefix.cc, namespaces reconciliation, vocabulary management

1 Introduction

RDF vocabularies bring their meaning to linked data by defining classes and properties, and their formal semantics. Relying on W3C standards RDFS or OWL, those vocabularies are a fundamental layer in the architecture of the Semantic Web. Without the explicit semantics declared in vocabularies, linked data, even using RDF, would be just linked pieces of information where links have no meaning. Interoperability between data and datasets rely heavily on shared vocabularies, but given the distributed nature of the Web, vocabularies are published by independent parties and there is no centralized coordination of this publication, nor should it be. Various independent services have been developed in order to discover vocabularies and provide information about them, and the community of data publishers and vocabulary managers have all interest

in complementarity and coordination between such services. In this paper, we focus on a specific aspect of vocabularies: their identification by namespaces and associated prefixes.

In the original XML syntax of RDF, prefixes are simply local shortcuts associated with XML namespaces using `xmlns` declarations. The usage of prefixes has been further extended to other syntaxes of RDF such as N3 and Turtle. Although a prefix to namespace association is syntactically limited to the local context of the file in which it is declared, common prefixes such as `rdf:`, `rdfs:`, `owl:`, `skos:`, `foaf:` and many more have become de facto standards. For example, RDFa has 1.1 has a default profile made of 11 well-used vocabularies based on their general usage on the Semantic Web according to the crawl of Yahoo! and Sindice as of March 2013[1]. Similarly, the YASGUI SPARQL editor has a list of built-in prefix-namespace associations to ease the construction of SPARQL queries. However, this list of "standard" prefixes is open-ended. Interfaces such as SPARQL endpoints (e.g. Virtuoso) use a list of built-in prefixes declaration for more and more namespaces but the choice of entries in this list is all but transparent. Hence, the reason of a given namespace being or not in this list could be interpreted in many ways, a potential source of technical and social conflicts. Therefore, the notion has been slowly spreading, at least implicitly, that common prefixes could and indeed should have a global use, implying some kind of governance and good practices. More and more vocabularies explicitly recommend the prefix that should be used for their namespace, generally using a common if not written good practice to avoid frontal clashes by recommending a prefix not already used. But there is no global policy except implicit rules of fair use to avoid potential conflicts resulting from polysemy (different namespaces using or recommending the same prefix) or synonymy (different prefixes used for the same namespace).

A vocabulary publisher needs to have access to some services capable of monitoring the existing prefixes usage in order to stick to those rules. In this paper, we focus on two services providing such information on prefixes usage namely prefix.cc[2] and LOV (Linked Open Vocabularies) [5]. Both services provide associations between prefixes and namespaces but following a different logic. The prefix.cc service allows anybody to suggest a prefix to namespace association. It supports polysemy and synonymy, and has a very loose control on its crowdsourced information. What it provides is more a measure of popularity of prefixes and namespaces than a way to put order in them. LOV has a much more strict policy forbidding polysemy and synonymy, enforced by a dedicated back-office database infrastructure, ensuring that each vocabulary in the LOV database is uniquely identified by a prefix, this unique identification allowing the usage of prefixes in various LOV publication URIs. This requirement leads sometimes to a situation where LOV uses prefixes different from the ones recommended by the vocabulary publishers.

[1] `http://www.w3.org/2010/02/rdfa/profile/data/`

[2] Service: `http://prefix.cc/`; Code: `https://github.com/cygri/prefix.cc`

The initial motivation of the work presented in this paper was to provide some kind of harmonization between those two services, from simple obvious tasks such as checking that prefix.cc provides all prefixes present in LOV and add them as necessary, to more complex ones such as detection and possible resolution of conflicts. We describe an approach for discovering new vocabularies in the wild by reconciling vocabularies in prefix.cc using SPARQL federated queries. This work was made semi-manually and involved collaboration between the two services managers to exchange data and take actions on each side. The remainder of this paper is structured as follows. In Section 2, we provide an overview of related work and services that support vocabulary management including the current approaches implemented by the LOV and prefix.cc services. In Section 3, we present how we have aligned those two services, detected conflicts and resolved them. In Section 4, we describe a method enabling to find new LOV-able vocabularies from the prefix.cc service. Finally, we discuss some lessons learned and outline future work in Section 5.

2 Related Work

Many different type of repositories exist to support users and developers to find controlled terms and entire vocabularies or ontologies on the web of data. We first describe the LOV initiative (Section 2.1) and we propose then our own classification based on the content, the domain, the purpose and the way such catalogs are populated or index created (Section 2.2).

2.1 Linked Open Vocabulary (LOV)

The Linked Open Vocabularies (LOV) initiative aims to bring more insights about published vocabularies in order to foster their reuse. Compared to other projects, LOV benefits from a community:

- to assess the quality (including documentation, metadata) and the reuse potential of a vocabulary before it is indexed. LOV contains currently 350+ reusable and well-documented vocabularies;
- to augment vocabularies with explicit information not originally defined in the RDF vocabulary. For example, only 55% of vocabularies have explicit metadata of at least one creator, contributor or editor. In LOV, we augmented this information leading to more than 85% of vocabularies with this information;
- to automatically extract the implicit relations between vocabularies using the Vocabulary Of Friend[3] (VOAF) ontology. These relations can be used as a new metric for ranking terms based on their popularity at the schema level;
- to consider vocabulary semantic in the result ranking: a literal value matched for the `rdfs:label` property has a higher score than for the `dcterms:comment` property.

[3] `http://lov.okfn.org/vocab/voaf/`

The way vocabularies are considered in LOV is similar to the way datasets are considered in the LOD cloud [2]. Hence, while the Vocabulary of Interlinked Datasets (VoiD) is used to describe relationships between datasets and their vocabularies [1], VOAF is used to describe the mutual relationships between vocabularies. VOAF itself reuses over popular vocabularies such as Dublin Core Terms (dcterms), Vocabulary Of Interlinked Datasets (VoiD), Vocabulary for ANNotating vocabulary (vann) and the BIBliographic Ontology (bibo). The vocabulary also introduces new classes such as `voaf:Vocabulary` and `voaf:VocabularySpace`.

The LOV-Bot is the tool that automatically keeps up-to-date the relationships and the metadata about the vocabularies indexed in LOV, using the following steps:

- LOV-Bot daily checks for vocabularies update (any difference in the vocabulary formal description fetched using content negotiation);
- LOV-Bot uses SPARQL constructs to detect relationships and metadata and creates explicit metadata descriptions in the LOV dataset;
- LOV-Bot annotations are then listed in a back-office administration dashboard in order to be reviewed. This manual part enables LOV curators to interact with vocabularies authors and the wider community to raise issues and make remarks or suggestions.

The LOV dataset is synchronized with the information presented in the web site. The latter allows a human user to browse LOV information. The Linked Open Vocabularies initiative does not only monitor the current state of the ecosystem. It also aims at storing and giving access to vocabularies history. To achieve this goal, the LOV database contains every different version of a vocabulary over the time since its first issue. For each version, a user can access the file and a log of modifications since the previous version.

2.2 Ontology Repositories

While we refer the reader to [3] for a systematic survey of ontology libraries, we give our own classification of ontology repositories (Table 1). In particular, we distinguish six categories of catalogs:

- *Catalogs of generic vocabularies/schemas* similar to the LOV catalog, but without any relations among the vocabularies. Example of catalogs falling in this category are vocab.org[4], ontologi.es[5], JoinUp Semantic Assets or the Open Metadata Registry.
- *Catalogs of ontologies for a specific domain* such as biomedicine with the BioPortal[6], geospatial ontologies with SOCoP+OOR[7], Marine Metadata Interoperability and the SWEET ontologies[8].

[4] `http://vocab.org/`
[5] `http://ontologi.es/`
[6] `http://bioportal.bioontology.org/`
[7] `http://socop.oor.net/`
[8] `http://sweet.jpl.nasa.gov/2.1/`

– *Catalogs of ontologies from a project* such as the famous DAML repository of ontologies[9].
– *Catalogs of ontology Design Patterns (ODP)* focused on reusable patterns in ontology engineering.
– *Catalogs of editors' ontologies* used to test some features of a tool and to keep track of the ontologies built by a tool, such as Web Protégé or TONES.
– *Catalogs of ontologies maintained by a single organization* which often uses a platform such as Neologism[10] for publishing vocabularies.
– *Vocabularies crawled by Semantic Web search engines* containing snapshots at the time of the crawsuch as Watson[11], Sindice[12], Falcon-s[13] or Swoogle.

Catalog name	Number of vocabularies	Search Feature	Category	Vocabulary maintenance
vocab.org	19	No	Catalog of generic vocabularies	N/A
ontologi.es	39	No	-//-	N/A
Joinup Semantic Assets	112	Yes	-//-	Yes
Open Metadata Registry	308	Yes	–//–	Yes
BioPortal	355	Yes	Catalog of Domain vocabularies	Yes
SOCoP + OOR	40	Yes	–//–	Yes
Marine Metadata Interoperability	55	Yes	–//–	Yes
SWEEET 2.2	200	No	–//–	N/A
DAML	282	No	–//–	No
ODPs	101	No	Catalog of ODPs	Yes
vocab.derie.ie	68	No	Catalog of Organizations	Yes
data.lirmm.fr ontologies	15	No	–//-	Yes
TONES	219	No	Catalog of editors' vocabularies	N/A
Web Protégé	69	No	–//–	Yes

Table 1. Catalogs of vocabularies with respectively the number of the ontologies, the presence of a search feature, the catalog category and whether it is maintained or not

[9] http://daml.org/ontologies/
[10] http://neologism.deri.ie
[11] http://watson.kmi.open.ac.uk/
[12] http://www.sindice.com
[13] http://ws.nju.edu.cn/falcons/

We observe that the existing catalogs of vocabularies in the literature have some limitations compared with LOV. In terms of coverage, the number of vocabularies indexed by LOV is constantly growing and it is the only catalog, to the best of our knowledge, that provide all types of search criteria (metadata search, within/across ontologies search), both an API and a SPARQL endpoint access and that can be as well classified as an "Application platform" apart from being at the same time an ontology directory and an ontology registry. According to the categories of ontology libraries defined in [3], LOV falls under the category of "curated ontology directory" and an "application platform" because the ontologies are curated manually with statistics automatically generated, and because it exposes its data via an API. Furthermore, LOV provides an answer to some of the issues mentioned in the survey reported in [3], such as "where has an ontology been used before?" or "is this ontology compatible with mine?". In particular, LOV provides vocabulary usage statistics of the LOD Cloud datasets and it exposes vocabularies dependency using the Vocabulary-of-A-Friend (VOAF) ontology.

vocab.cc[14] is a service which is similar to prefix.cc since it enables to look up and search for Linked Data vocabularies while providing more specific information about the usage of a particular class or property in the Billion Triple Challenge Dataset (BTCD). It also provides the ranking of those properties or classes. The authors mentioned that "common prefixes are resolved with data from prefix.cc". Although they don't give further details, this service is somehow related to prefix.cc. Triple-Checker[15] is a web service based on prefix.cc which aims at finding typos and common errors in RDF data. It parses a given URI/URL and the output is divided in two sections: the namespaces and the term section. The former matches against prefix.cc to determine whether they are "common prefixes" and the latter provides the term definition.

3 Aligning LOV with Prefix.cc

In this section, we present how we perform the alignment between the two services LOV and prefix.cc. Figure 1 shows the evolution of the number of prefixes registered in these two services between April 2009 and July 2013. Our main goals are to align Qnames (prefix) to a unique URI in LOV and to make sure that all the vocabularies in LOV are actually inserted in prefix.cc.

We propose to perform SPARQL queries over all the files of prefix.cc at `http://prefix.cc/popular/all.file.vann` in the FROM clause and compare them to the content of the LOV SPARQL endpoint[16] via a SERVICE[17] call. The SERVICE keyword defined in the SPARQL 1.1 Query Language instructs a federated query processor to invoke a portion of a SPARQL query against a remote SPARQL endpoint [4]. Results are returned to the federated query

[14] `http://vocab.cc`

[15] `https://github.com/cgutteridge/TripleChecker`

[16] `http://lov.okfn.org/endpoint/lov`

[17] `http://www.w3.org/2009/sparql/docs/fed/service`

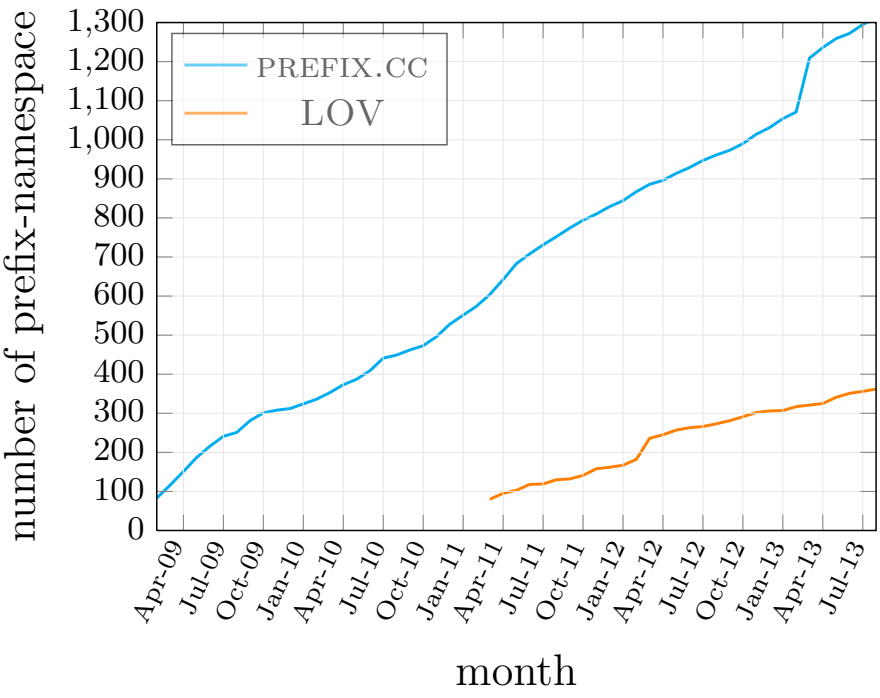

Fig. 1. Evolution of the number of prefix-namespace pairs registered in prefix.cc and LOV

processor and are combined with results from the rest of the query. To be more generic and standards-compliant, the queries could be run with the Jena ARQ command-line tool to produce a CSV or a JSON serialization that could be easily consumed either by the prefix.cc backend via phpMyAdmin or by the LOV backend.

3.1 First Task: prefixes in LOV not present in Prefix.cc

First, we compute $< LOV > INTERSECTS < PREFIX.CC >$ and $< LOV > MINUS \{< LOV > INTERSECTS < PREFIX.CC >\}$. The following SPARQL query finds namespace URIs in LOV that do not exist in prefix.cc along with their LOV prefix.

```
PREFIX vann: <http://purl.org/vocab/vann/>
SELECT ?prefix ?lovURI
FROM <http://prefix.cc/popular/all.file.vann> {
  SERVICE <http://lov.okfn.org/endpoint/lov> {
    SELECT ?prefix ?lovURI {
```

```
      [] vann:preferredNamespacePrefix ?prefix;
        vann:preferredNamespaceUri ?lovURI;
    }
  }
  FILTER (NOT EXISTS { [] vann:preferredNamespaceUri ?lovURI })
  OPTIONAL {
    [] vann:preferredNamespacePrefix ?prefix;
      vann:preferredNamespaceUri ?pccURI;
  }
}
ORDER BY ?prefix
```

The first results[18] shown the following: $card(LOV) \bigcap card(PREFIX.cc) = 188$[19] and $card(LOV) - card(PREFIX.cc) = 133$[20] prefixes in LOV not yet registered in prefix.cc. At this point, a first batch of 80 prefixes/namespaces from LOV were safely imported in prefix.cc since there were no conflicts. For the remaining conflicting ones, they needed more in-depth analysis.

3.2 Second Task: Dealing with Conflicts between Prefix.cc and LOV

In the process of alignment, there were two types of conflicts and we provide appropriate actions and/or solutions accordingly:

- Clashes: cases where we have in both services the same prefix but different URIs;
- Disagreements on preferred namespace: cases where for the same URI, we found different prefixes.

Clashes. We performed a SPARQL query as above to identify clashes in vocabularies (30). In Table 2, we identify seven different types of issues to deal with, such as (i) real conflicts, (ii) URIs are 404, (iii) URIs are obsolete versions and (iv) two URIs redirecting to the same resource.

Disagreements on namespace URIs. The general idea is that if vocabulary editors have not included explicitly a `vann:preferredNamespacePrefix` in their description, the curators of LOV are free to change it and put whatever seems appropriate. At the same time, in prefix.cc, having multiple prefixes for the same namespace IRI in not a problem. However, we computed those prefixes in LOV that have different prefixes in prefix.cc. The following query retrieves the URIs falling in those disagreements:

[18] This query was performed in two weeks between March, 2nd and March, 20th 2013 and at this time, card(LOV) = 321 vocabularies while card(Prefix.cc) = 925

[19] http://www.eurecom.fr/~atemezin/iswc2013/experiments/firstAlignments/intersection-prefixLOV-02-03.csv

[20] http://www.eurecom.fr/~atemezin/iswc2013/experiments/firstAlignments/inLovNotINPrefixcc-02-03.csv

Type of issue	# Vocabularies	Percentage
pccURI and lovURI redirect to same resource	8	26.67%
lovURI already in prefix.cc as secondary	7	23.3%
Real conflicts	6	20%
pccURI is 404	4	13.3%
pccURI is an obsolete version	3	10%
lovURI is 404	1	3.3%
lovURI is an obsolete version	1	3.3%

Table 2. Type of issues encountered for vocabulary clashes

```
PREFIX vann: <http://purl.org/vocab/vann/>
SELECT ?prefix ?lovURI ?prefixcc
FROM <http://prefix.cc/popular/all.file.vann> {
  SERVICE <http://lov.okfn.org/endpoint/lov> {
    SELECT ?prefix ?lovURI {
      [] vann:preferredNamespacePrefix ?prefix;
        vann:preferredNamespaceUri ?lovURI;
    }
  }
  FILTER (?pccURI = ?lovURI && ?prefix != ?prefixcc)
  OPTIONAL {
    [] vann:preferredNamespacePrefix ?prefixcc;
      vann:preferredNamespaceUri ?pccURI;
  }
}
ORDER BY ?prefix
```

From the results of this query (61 cases), we have three actions to perform:

- add the lovPrefix (prefix in LOV) in prefix.cc (e.g: adding `geod:http://vocab.lenka.no/geo-deling#`) to the existing `ngeoi` in *pccPrefix*.)
- add more alternative URIs to the existing prefix in prefix.cc (e.g: adding `prov:http://purl.org/net/provenance/ns#`) to the existing `hartigprov, prv` in *pccPrefix*)
- change a prefix in LOV[21] (e.g: lovPrefix `dc` for `http://purl.org/dc/terms` not in the list $\{dcterm, dcq, dct, dcterms\}$ has been replaced by `dce` in LOV).
- No changes when the lovPrefix is contained in the set of prefixes of prefix.cc.

3.3 Social Aspects

Several vocabularies are maintained by a community of users. As part of the alignment process, we contacted the authors, creators or maintainers (if they

[21] `http://www.eurecom.fr/~atemezin/iswc2013/material/action-sameUriDifferentPrefixes.pdf`

exist) of vocabularies to involve them as well in the process of changing prefixes, and agree with them to fix some issues regarding their vocabularies. From the homepages of the vocabulary authors and editors collected in LOV, we connect to their social platform accounts such as LinkedIn, Google+ or Twitter. Table 3 summarizes some cases of real conflicts where the LOV curators have to find and contact the editors of the vocabularies for negotiation.

prefix	lovURI	Remark
sp	http://data.lirmm.fr/ontologies/sp#	contact editor at LIRMM ($sp \Rightarrow osp$)
scot	http://scot-project.net/scot/ns#	contact editors at lovURI
media	http://purl.org/media#	contact editors for negotiation
pro	http://purl.org/spar/pro/	contact editors for negotiation
swp	http://www.w3.org/2004/03/trix/swp-1/	contact editors, fix on LOV side
wo	http://purl.org/ontology/wo/core#	contact editors
idemo	http://rdf.insee.fr/def/demo#	to resolve with INSEE

Table 3. LOV and prefix.cc conflicts resolution leading to contact vocabularies editors for negotiation. We provide the prefix, the URI in LOV and the action undertaken.

4 Finding Vocabularies in Prefix.cc

We want to find out in prefix.cc, which of the couples (prefix, URI) could be potentially a vocabulary to be further assess to be included in the LOV catalog. To address this question, we first compute all the differences on prefix.cc NOT in LOV, i.e. $PREFIX.CC$ MINUS ($LOV < INTERSECT > PREFIX.CC$), performing using a SPARQL query. This results in 742 URIs to be checked[22].

4.1 LOV Check API

We have implemented an API[23] that allows a user to run the LOV-Bot over a distant vocabulary. It takes as parameter the vocabulary URI to process and the time out (integer) specified to stop the process. The result of this action is a set of 26 property-values from which we are interested in using only 8 of them, namely:

- **uri** (string) – uri of the vocabulary.
- **namespace** (string) – namespace of the vocabulary.
- **prefix** (string) – prefix of the vocabulary
- **inLOV** (boolean) – indicates if the vocabulary is already in the Linked Open Vocabularies ecosystem.

[22] http://www.eurecom.fr/~atemezin/iswc2013/experiments/input/notInLOV.json

[23] http://lov.okfn.org/dataset/lov/apidoc/

– **nbClasses** (int) – Number of classes defined in the vocabulary namespace.
– **nbProperties** (int) – Number of properties defined in the vocabulary namespace.
– **dateIssued** (string) – Vocabulary date of issue.
– **title** (Taxonomy) – List of titles with language information if available.

The code below gives the response of our algorithm for the vocabulary identified at `http://ns.aksw.org/Evolution/`.

```
[caption={Sample output of a response of the Check API}]
{
 "dateIssued": "None",
 "inLOV": false,
 "namespace": "http://www.agfa.com/w3c/2009/clinicalProcedure#",
 "nbClasses": 47,
 "nbProperties": 29,
 "pccURI": "http://www.agfa.com/w3c/2009/clinicalProcedure",
 "prefix": "clinproc",
 "title": [
   {
     "dataType": null,
     "language": "en",
     "value": "Clinical Procedure"
   }
 ],
 "uri": "http://www.agfa.com/w3c/2009/clinicalProcedure"
},
```

4.2 Experiments

We wrote a script calling the LOV Check API on the URIs in prefix.cc for determining the candidates vocabularies to be inserted in LOV using the algorithm in Listing 1. We ran four times the experiments (possibly due to some network instabilities) in order to determine from which results what should be assessed. Table 4 gives an overview of the number of URIs with respectively the attribute "inLOV=false"(TP), "inLOV=true"(FP) and the errors occurred (Null returned, http/proxy or time out reached by the API).

Regarding the experiments, `Experiment4` gives stable results with less network errors. Therefore, we stick on this experiment to report our findings and analysis. We found that 227 (43.48%) are vocabularies in the sense of LOV since they have at least one property or one class. 297 vocabularies (56.51%) might have some problems (or are even not vocabularies at all) as they have neither classes nor properties. Regarding the presence of prefixes, we found 140 (61.67%) of them. The 227 vocabularies could all be inserted in the LOV catalog since they fulfill the current requirements of what is a " LOV-able vocabulary". In this list, we found vocabularies such as `rdf, rdfs, owl` that are used to build other vocabularies but are not yet integrated in the LOV catalog.

	TP(inLOV=false)	FP(inLOV=true)	Errors
Experiment1	525	44	173
Experiment2	403	26	313
Experiment3	351	28	363
Experiment4	522	44	176

Table 4. Experiments looking for stable results of finding vocabularies in prefix.cc.

Algorithm 1 finding vocabularies NOT in LOV from prefix.cc algorithm

Open $notInLOV.jsonfile$ containing the prefix.cc URIs **not** in LOV
initialize $item$ as List
Initialize $result$ as collection of $item$
for each $pccURI \in notInLOVfile$ **do**
  $uri \leftarrow$ value of $pccURI$
  $uriv \leftarrow$ construct-valid uri
  call LOV-Check API with parameter $uriv$
  try/catch HTTPError, URLError, IOError, ValueError
  while no error raised **do**
    initialize $item$ to an empty List
    append $pccURI, prefix, inLOV, namespace, title, dateIssued, nbClasses, nbProperties$ in $item$ List
    append $item$ to $result$
  end while
end for
print $output - result$

From the list of URIs that were not LOV-able vocabularies, we wanted to do more analysis by checking the RDF files using the Triple-Checker tool. Our aim is to be sure if we did not leave out some candidate vocabularies or if there are other type of errors such as parsing errors. Table 5 provides results classified into 4 categories:

- General errors such as loading files or proxy errors: 78.30%
- Candidate LOV-able vocabularies: 12.20%
- Clearly not vocabularies (`nbClasses` = `nbProperties` = 0), typically instances, datasets, html pages: 6.45%
- Others (mainly parsing errors): 3.05%

5 Conclusion

In this paper, we have analyzed numerous vocabularies referenced in LOV and in prefix.cc and we have presented a way to manage the prefixes of those vocabularies. We have shown that in the process of mapping namespaces with prefixes, some conflicts have to be resolved, often by contacting the editors themselves.

Total URIs	295	100%
Loading/404 errors	182	61.69%
Vocabularies	36	12.20%
Proxy errors	27	9.15%
50x, 40x errors	22	7.45%
Parsing errors	9	3.05%
Web Pages containers	9	3.05%
No triples found	8	2.71%
RDF data	2	0.67%

Table 5. Analysis of the URIs with no classes and no properties while using the LOV-Bot API

One future work is to develop a new strategy for the LOV-Bot API to take into account vocabularies published in other formats such as n3 and turtle. This would require to first test the validity of those formats and to adapt the way namespaces are obtained in order to not check only the presence of the `vann:preferredNamespace` property but to rely on similarity algorithm in order to guess the closest namespace given a URI vocabulary and some statistics of the number of classes and properties.

The work presented in this paper can be extended in several directions. Sticking to the two services we have studied and already contributed to harmonize, the possible next steps would be to automate as far as possible the tasks that have been made semi-automatically so far: *i)* developing a unique interface for submitting namespaces and prefixes to both services; *ii)* bridging the LOV back-office and the prefix-cc database using both services API in order to publish a list of common recommended prefixes. The latter goes beyond the limited framework of the two original services since such a list could be consolidated and endorsed by the main actors in vocabulary publication and management, and recommended for use in linked data applications. This could be picked up by the upcoming W3C Vocabulary Management Working Group as part of the new Data Activity[24].

This (apparently) simple issue of prefixes and namespaces is providing a good illustration of why some kind of governance is needed in the distributed ecosystem of vocabularies and linked data, pointing to both technical and social aspects, and proposing concrete examples of conflict resolution. There is no, and certainly there should never be any, central attribution authority for prefixes, and the needed regulation has to be made a posteriori, including good practices of cooperation and negotiation between vocabulary publishers. Development and harmonization of services such as LOV and prefix.cc is then to be considered as

[24] `http://www.w3.org/2013/05/odbp-charter.html`

part of the current and more general effort already started by the DCMI[25] and W3C[26] for a sustainable governance of vocabularies.

Acknowledgments

This work is partially supported by the project Datalift funded by the French Research Agency (ANR) under grant number ANR-10-CORD-009. The Linked Open Vocabularies initiative is hosted by the Open Knowledge Foundation. The authors are very grateful for the support and help of Richard Cyganiak, author and maintainer of the prefix.cc service.

References

1. K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao. Describing linked datasets. In *2nd Workshop on Linked Data on the Web (LDOW)*, Madrid, Spain, 2009.
2. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
3. M. DÁquin and N. Noy. Where to publish and find ontologies? a survey of ontology libraries. *Web Semantics: Science, Services and Agents on the World Wide Web*, 11(0):96–111, 2012.
4. E. Prud'hommeaux and C. Buil-Aranda. SPARQL 1.1 Federated Query. W3C Recommendation, 2013. http://www.w3.org/TR/sparql11-federated-query/.
5. B. Vatant and P.-Y. Vandenbussche. Catalogue de Vocabulaires. Datalift, D2.2, 2013. http://datalift.org/en/node/18.

[25] Long-term Preservation and Governance of RDF Vocabularies: http://dcevents.dublincore.org/IntConf/index/pages/view/vocPres

[26] W3C Vocabulary Services: http://www.w3.org/2013/04/vocabs/