Linked Web APIs Dataset

Web APIs meet Linked Data

Editor(s): Rinke Hoekstra, Universiteit van Amsterdam, The Netherlands
Solicited review(s): Christoph Lange, University of Bonn, Germany; Enrico Daga, KMI, The Open University, UK; Tobias Kuhn VU Amsterdam, The Netherlands

Milan Dojchinovski and Tomas Vitvar
Web Intelligence Research Group, Faculty of Information Technology, Czech Technical University in Prague
E-mail: {milan.dojchinovski,tomas.vitvar}@fit.cvut.cz

Abstract.
Web APIs enjoy a significant increase in popularity and usage in the last decade. They have become the core technology for exposing functionalities and data. Nevertheless, due to the lack of semantic Web API descriptions their discovery, sharing, integration, and assessment of their quality and consumption is limited. In this paper, we present the Linked Web APIs dataset, an RDF dataset with semantic descriptions about Web APIs. It provides semantic descriptions for 11,339 Web APIs, 7,415 mashups and 7,717 developer profiles, which make it the largest available dataset from the Web APIs domain. The dataset captures the provenance, temporal, technical, functional, and non-functional aspects. In addition, we describe the Linked Web APIs Ontology, a minimal model which builds on top of several well-known ontologies. The dataset has been interlinked and published according to the Linked Data principles. Finally, we describe several possible usage scenarios for the dataset and show its potential.

Keywords: Web APIs, Linked Data, Web services, Linked Web APIs, ontology

1. Introduction

Web APIs have become the first-class citizens on the Web and the core functionality of any Web application. Targeting the developer audience, they lower the entry barriers for accessing valuable enterprise data and functionalities. Back in late 2008, ProgrammableWeb.com[1], the largest Web API and mashup directory, reported only 1,000 Web APIs. This count increased to 5,000 APIs in Feb 2014 and over 13,000 APIs in June 2015. There are several benefits of having these Web API descriptions provided as Linked Data. The Web API descriptions are contextualized, they can be referenced, re-used and combined. The Web APIs data is linked, therefore API consumers can effectively discover new Web APIs. Last but not least, on the one hand the developers can benefit from sophisticated discovery and selection queries for discovery and selection of APIs, on the other hand the Web API providers can execute queries to get better insight and analysis results from the Web APIs ecosystem.

In order to achieve these goals, we have developed the *Linked Web APIs* dataset. It provides information about Web APIs, mashups which utilize Web APIs in compositions, and mashup developers. The primary source for the dataset is the ProgrammableWeb.com directory, which acts as central repository for Web APIs descriptions. The dataset re-uses several well-known ontologies developed by the Semantic Web community. In order to conform to the Linked Data principles[2], we have also linked the dataset with four

[1] `http://www.programmableweb.com`

[2] `http://www.w3.org/DesignIssues/LinkedData.html`

central LOD datasets: DBpedia[3], Freebase[4], LinkedGeoData[5] and GeoNames[6].

The remainder of this paper is structured as follows. Section 2 describes the source of information and how the data was collected. Section 3 describes the developed ontology for modelling relevant Web APIs information. The creation of the Linked Web APIs dataset and its technical details are described in Section 4. The approach for interlinking the dataset with other LOD datasets is described in Section 5. Section 6 discusses the quality of the ontology and the dataset. Section 7 presents selected use cases and the results from a survey on the potential and the usefulness of the dataset. Section 8 discusses related vocabularies and potential data sources. Finally, Section 9 concludes the paper.

2. The Data Source

In our work, we have considered ProgrammableWeb as a primary source of information for creating the dataset. It adopts characteristics of a social Web platform where Web API providers can publish and share information about offered Web APIs and consequently increase their visibility. The API directory also allows developers to find appropriate APIs for their projects, or they can learn from showcases of existing mashup applications.

The implemented knowledge extraction process consists of four steps: (1) parsing and extraction of valuable information from pages describing Web APIs, mashups and developers, (2) pre-processing, cleanup and consolidation of information, (3) linking with LOD resources, and (4) lifting in RDF and publishing the data as Linked Data.

An example of a Web page which describes a Web API is the one which describes the Twitter API[7]. For each Web API we extracted its title, short summary describing its functionalities, assigned tags and categories, technical information, such as supported formats and protocols, as well as non-functional properties such as its homepage, usage limits, usage fees, security, etc. Similarly, for each mashup we extracted its title, short free-text description of its functionalities, assigned tags and the homepage. From each page which describes a developer, we extracted its username, homepage and short bio. The city and country of residence, its given and family name and the gender were extracted only if these information were publicly available.

We also captured the *relationships* between the Web APIs, mashups and developers. In other words, for each mashup we extracted the list of Web APIs which were used by the mashup and also the list of mashups created by each developer. The dataset also captures the *temporal aspects* - the creation time of the Web APIs, mashups and the time a user registered his profile.

In order to collect the data, we have implemented a script which systematically browses and parses relevant pages. The parsing mechanism has been implemented using the jsoup Java HTML parser[8]. For the crawler, we used proper etiquette and we configured the crawl delay to one page every four seconds.

3. The Ontology

The Linked Web APIs ontology[9] is a minimal model that captures the most relevant information which are related to Web APIs and mashups. The ontology builds on top of existing and well established ontologies and appropriately extends them. The selection of the ontologies was driven by the following four crucial requirements:

- *Provenance:* It is important to keep information about *Who* (developers) created *What* (mashups) and *How* (using which APIs). In addition, information about APIs a provider provides need to be captured as well.
- *Functional and Non-functional Properties: What* functionalities a Web API or mashup offers is more than important, as well as their usage limits and fees, supported security and authentication mechanisms.
- *Technical Properties:* Information about the supported protocols, formats and the Web API endpoint location is as important, as it allows a Web API consumer to search only for APIs with preferred technical capabilities.

[3] http://dbpedia.org/

[4] https://www.freebase.com/

[5] http://linkedgeodata.org/

[6] http://www.geonames.org/

[7] http://www.programmableweb.com/api/twitter

[8] http://jsoup.org/

[9] http://linked-web-apis.fit.cvut.cz/ns/core/index.html

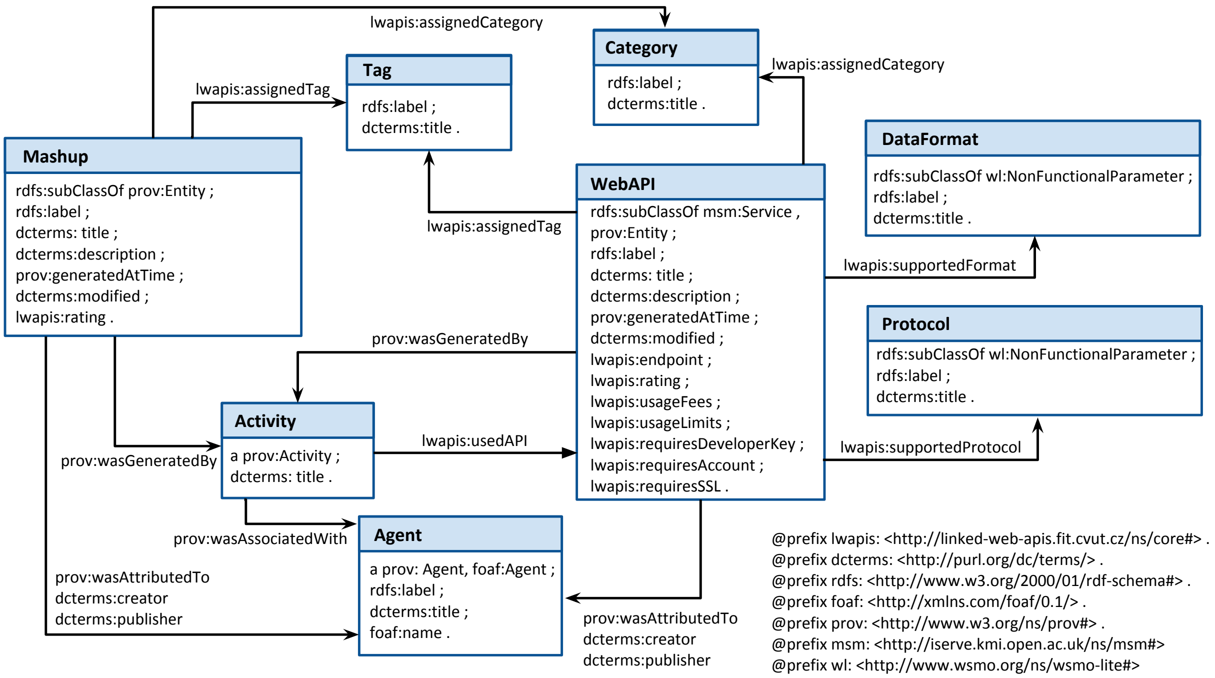

Fig. 1. The Linked Web APIs Ontology.

- *Temporal Information: When* a mashup or Web API was created is a valuable information as well. For example, to analyze the recent trends in the API ecosystem or to discover the most recent Web APIs or mashups.

Figure 1 shows the overall Linked Web APIs ontology. The ontology contains three central classes: *lso:WebAPI* – describe Web APIs, *lso:Mashup* – describe mashup compositions, which utilize one or more Web API, and *lso:Agent* – represent all kinds of entities involved in the creation and/or consumption of Web APIs and mashups.

In order to capture the provenance information, the Linked Web APIs ontology integrates the PROV-O ontology[10] by incorporating the classes *prov:Entity*, *prov:Activity* and *prov:Agent*, and their related properties. The *prov:Entity* class serves as super-class of *lso:WebAPI* and *lso:Mashup* classes. The activities convey information about the consumption process of the Web APIs and the generation of mashups by the agents. Note that an activity can also refer to the action of the creation of an API documentation (i.e., a ProgrammableWeb entry) and this can be modeled by associating an action with an instance of the *hydra:ApiDocumentation* class from the Hydra[11] vocabulary. We introduce the *lso:usedAPI* property which refines the semantics of the *prov:used* property so it can be used to explicitly identify usage of a Web API in a mashup creation. The temporal information about the creation time of a mashup or Web API is expressed using the *prov:generatedAtTime* property. It represents the time when the API documentation has been generated withing the snapshot. Since this is the first official snapshot of the dataset, the value of this property indicates the first version of each resource. Any changes in the following snapshots will be captured with the *dcterms:modified* property. Also note that in the most cases the *prov:generatedAtTime* is the time when the API has been registered at ProgrammableWeb. For example, the Google Maps API, the Twitter API and the YouTube API, are between the most popular APIs at ProgrammableWeb, and the provided time information are justifiable; May 5th, 2005 for the Google Maps API, August 12th, 2006 for the Twitter API, and August 6th, 2006 for the YouTube API.

For the functional (tags and categories) and nonfunctional (formats and protocols) properties of the Web APIs and mashups we introduce new classes

[10] http://www.w3.org/TR/prov-o/

[11] http://www.hydra-cg.com/spec/latest/core/

in our namespace. The ontology also integrates the *wl:NonFunctionalParameter* class from the WSMO-lite ontology[12] [9], which was developed by the Semantic Web Services community, to explicitly identify non-functional properties. The Minimal Service Model (MSM)[13] ontology, which was initially defined for the hRESTS microformat [7], is also considered. The class *msm:Service* is integrated as super-class of the *lso:WebAPI*. This allows us to attach additional Web API information, such as operations, inputs and outputs, which is relevant for the execution of Web APIs. General metadata information such as a Web API and mashup title, or their short textual description are described using the Dublin Core vocabulary[14].

4. The Linked Web APIs Dataset - Coverage, Availability and Maintenance

4.1. Coverage and Availability

The Linked Web APIs dataset is the first of its kind with descriptions for more than 11,339 Web APIs, 7,415 mashups and 7,717 mashup creators. Overall, the dataset contains over 550K RDF triples. For all the resources we mint URIs in our own namespace (`http://linked-web-apis.fit.cvut.cz/resource/{name}`). The `name` part from the URIs is a normalized form of the resource label, which is lowercased and each space is replaced with an underscore sign. Since it is possible that two different resources can have the same name (e.g., the label `XML` can occur as a tag and also as a format), we attach its type as a suffix to the URI. For example, `_tag` for tags or `_api` for Web API URIs. The URI `http://linked-web-apis.fit.cvut.cz/resource/google-maps_api` is an example of a URI minted for the Google Maps API. A similar approach is employed by DBpedia and Wikipedia[15] to distinguish pages which have the same title. For example, `/resource/Food_(band)` for a page describing the musical band "Food" and `/resource/Food_(film)` for a page describing the movie with the same name. It would be possible to use an alternative schema for the URIs, where the resource type is encoded as a path component (e.g. `/resource/apis/google-maps`). Which URI schema is more appropriate is a debatable question. Nevertheless, both approaches are valid and serve their purpose.

All the URIs are dereferenceable and served according to the Linked Data principles in RDF/XML and Turtle format. The dataset is available via a Virtuoso SPARQL endpoint and as an RDF dump. The landing page for the dataset is `http://linked-web-apis.fit.cvut.cz/` and it provides information about the latest news, releases and changes. Technical details about the dataset are listed in Table 1.

Table 1
Details of the Linked Web APIs dataset.

Name	Linked Web APIs dataset
URL	`http://linked-web-apis.fit.cvut.cz/`
Endpoint	`http://linked-web-apis.fit.cvut.cz/sparql`
Ontology	`http://linked-web-apis.fit.cvut.cz/ns/core#`
Version	0.1
Ver. Date	05.08.2015
License	Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)
Datahub	`https://datahub.io/dataset/linked-web-apis`

Currently, we employ the same versioning approach as the one used by DBpedia - versioning at the dataset level. Nevertheless, versioning at the resource level will be considered in the near future. Versioning at the resource level would be appropriate when integrating the APIs.io[16] repository (see Section 8.2 for more details), since the API versioning information is explicitly present.

4.2. Maintenance and sustainability

The computer center of the Czech Technical University in Prague kindly provided us with persistent web space for the publication of the dataset and the ontology. This will guarantee persistent URI identifiers for the dataset resources.

The ongoing maintenance of the dataset is carried out at the data level, as well as at the ontology level and its alignment with relevant existing and emerging vocabularies.

Our long-term goal is to establish the Linked Web APIs as a central Linked Data hub for Web API descriptions. To this end, we aim at providing support for various Web API description models (cf. Section 8.1

[12] `http://www.wsmo.org/ns/wsmo-lite/`

[13] `http://iserve.kmi.open.ac.uk/ns/msm#`

[14] `http://purl.org/dc/terms/`

[15] `https://en.wikipedia.org/wiki/Wikipedia:Article_titles#Disambiguation`

[16] `http://apis.io/`

and data sources with relevant Web API information (cf. Section 8.2).

It takes over 29 hours (with four seconds crawl delay) to complete the crawling, information extraction and RDF generation process. The process is fully automated and currently it has to be manually triggered. The dataset has been already integrated with DBpedia (via *owl:sameAs* links; see Section 5) and we plan to synchronize the Linked Web APIs dataset generation with the DBpedia releases and generate the dumps on bi-annual basis. Since new APIs are published every day, we also plan to provide a live extraction service which pulls updates in real-time and updates the triple store.

5. Dataset Linking

In order to assure maximal reusability and integrability, we linked the dataset with four central LOD datasets. Two multi-domain datasets, DBpedia and Freebase, and two geographical datases, GeoNames and LinkedGeoData. From the information we linked the Web APIs supported data formats, supported protocols, developers' city and country of residence. Since GeoNames and LinkedGeoData are geographical datasets, only users' city and country of residence were linked to those datasets. DBpedia and Freebase are multi-domain datasets and therefore we linked all information to these datasets. The links to DBpedia, and respectively to Freebase, were generated following the most-frequent-sense based approach which is used as entity linking method in the Entityclassifier.eu NER system [3]. The linking to LinkedGeoData was governed by the intuition that the names of the cities and countries in our dataset have same names in the LinkedGeoData dataset. The approach was supported by SPARQL queries which retrieve resources with a given label. Following this linking methodology we generated 1,440 links out of which 722 are DBpedia links, 299 Freebase links, 326 GeoNames links and 93 LinkedGeoData links. Table 2 provides more information about the linking.

We opted for these linking approaches, since we have the tooling in place and they served their purpose.

It is important to note that our dataset has also received in-links[17] from DBpedia, the most prominent LOD dataset. The links have been accepted and picked up with the latest DBpedia release (October 2015).

Table 2
Number of linked resource per type and dataset.

	DBpedia	Freebase	LGD	GeoNames
Formats	283	208	/	/
Protocols	123	91	/	/
Cities	263	/	47	276
Countries	53	/	46	50
Total	722	299	93	326

6. Quality

According to the 5-star classification system [1], defined by Tim Berners-Lee, the Linked Web APIs classifies as a five-star dataset. The five stars are credited for the open license, availability in a machine-readable format, use of open standards, use of URIs for identification, and the links to the other LOD datasets.

6.1. Dataset Quality

Zaveri et al. [11] provide a list of indicators for evaluation of the intrinsic quality of Linked Data datasets. We have checked the data for each of the metrics (where applicable) described for *syntactic validity* and *semantic accuracy*.
Syntactic validity: no errors were found when checked for syntax errors, syntactically accurate values and malformed literals. Raptor[18] RDF syntax parsing and serializing utility was used to make sure the dataset is a syntactically correct RDF.
Semantic accuracy: checked whether the data values correctly represent the real world facts. To this end, we have randomly created a set of 100 triples and manually checked their validity. Only two triples representing tags have been spotted as invalid. Note that no invalid triples were spotted for the provenance, technical and non-functional information.

6.2. Vocabulary Quality

According to the 5-star vocabulary classification [6], the Linked Web APIs ontology credits four out of five stars: for the machine and human -readable information about the vocabulary (2 stars), it is linked to other vocabularies such as WSMO-lite and PROV-O (3 stars) and for the provided metadata information for the vocabulary (4 stars). The fifth star is credited for vocabularies which have been referenced by other vo-

[17] `http://downloads.dbpedia.org/2015-10/links/`

[18] `http://librdf.org/raptor/`

cabularies. However, the Linked Web APIs vocabulary has not been yet used and referenced by other vocabularies.

6.3. *Known Shortcomings*

The information extraction process is not entirely flawless due to its dependency on the HTML structure. From early 2012, when we created the initial snapshot of the dataset, until early 2016, the HTML has changed only two times, which is approximately every two years. Nevertheless, we are also considering other potential data sources, which will soon be integrated as part of the Linked Web APIs dataset. APIs.io and mashape marketplace[19] are the two most prominent data sources (see Section 8.2 for the discussion about datasources). We are currently working on an integration of APIs.io and soon it will be part of the Linked Web APIs dataset.

As for the ontology, some properties, such as "usageFees" and "usageLimits", are currently modeled as plain literals. The main reason for such a decision was the diversity of the possible values of these properties in the data. Very often these properties are expressed in a natural language, thus its modeling is a challenging task. In the future, if a data source provides data of a greater quality for these properties, we will appropriately extend the ontology.

7. Usefulness of the Dataset

7.1. *Use Cases*

The availability of a dataset with Web APIs descriptions in RDF can support various use cases, including, but not limited to personalised Web API provisioning, API ecosystem analysis and automated processing of Web API descriptions. In this section, we describe selected use cases and existing applications of the Linked Web APIs dataset.

Use case 1: Personalised Recommendations. The Linked Web APIs dataset contains links between the mashup and the developer resources, which can be used as a pertinent source of information for developing Web API recommendation methods. As an example, a user, who has already picked a Web API for his/her mashup, could search for other compatible Web APIs. Such a scenario can be supported with the SPARQL query from Listing 1 which returns the top 5 most used Web APIs.

```
SELECT ?api (COUNT(?api) as ?count)
WHERE {
  ?mashup prov:wasGeneratedBy ?activity.
  ?activity lso:usedAPI ls:google-maps_api .
  ?activity lso:usedAPI ?api .
  FILTER (!strends(str(?api),"google-maps_api"))
}
ORDER BY DESC(?count)
LIMIT 5
```

Listing 1: Top 5 most used Web APIs with Google Maps API.

In a different scenario, a developer could customize the search query to narrow down the results to Web APIs which support a particular data format (e.g., JSON) or APIs from a specific category (e.g., social, government, etc.).

In the context of personalised recommendations, the dataset has been recently employed in several works around personalised recommendation of Web APIs [4] and Linked Data resources [5]. The papers describe methods which accommodate the user preferences by analyzing their history. The method proposed in [5] recommends resources of interest for users with similar tastes. Both works focus on developing graph based algorithms on top of the Linked Web APIs dataset which utilize the provisioning information (who developed what), functional properties (tags and categories) and temporal information (when a mashup or an API was developed). The target audience in both methods are ultimately API consumers.

Use case 2: Support for Automated API Discovery, Composition and Orchestration. There are semantic models which provide mechanisms for automated Web service discovery, composition and orchestration. SADI [10] defines such a mechanism for fully automated processing and integration of Web services. In such scenarios, the Linked Web APIs dataset can be used as a relevant source of discovery for Web APIs for a composition workflows. Assuming a user composer already picked her/his favorite API(s), with a query which is similar to the one in Listing 1, then he can retrieve a list of candidate APIs. These candidate APIs can be further validated and added to the composition workflows.

Use case 3: Temporal Analysis. The dataset also captures the temporal aspect, i.e., the time when a mashup

[19] `https://market.mashape.com/`

or a Web API was developed. These kind of information can help Web APIs providers to get better insights about the recent developments and study the consumption of a Web API, or the whole Web API ecosystem over a period of time. The benefits from having temporal information can be illustrated with the SPARQL query from Listing 2.

```
SELECT COUNT(?mashup) as ?count
WHERE {
  ?mashup prov:wasGeneratedBy ?activity.
  ?activity lso:usedAPI ls:google-maps_api .
  ?mashup prov:generatedAtTime ?date .
  FILTER (?date >= "2013-01-01"^^xsd:dateTime
    && ?date < "2014-01-01"^^xsd:dateTime)
}
```

Listing 2: Number of mashups utilizing the Google Maps API in 2013.

The SPARQL query in the listing gives information about the total number of mashups which utilized the Google Maps API in 2013. Figure 2 visualizes the results of the analysis for three popular APIs and their utilization over a period of time.

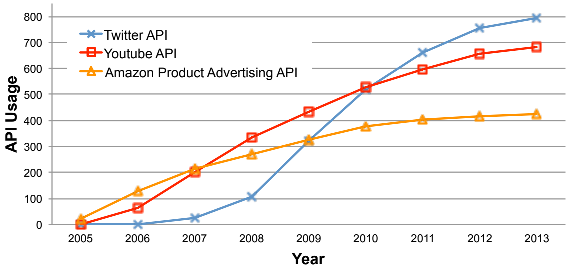

Fig. 2. Web API utilization over time.

The Web API provider might be also interested to find out in what kind of mashups their API was used. An answer to such a question can be answered with the SPARQL query in Listing 3.

```
SELECT ?category (COUNT(?category) as ?count)
WHERE {
  ?mashup prov:wasGeneratedBy ?activity.
  ?activity lso:usedAPI ls:google-maps_api .
  ?mashup lso:assignedTag ?category .
} ORDER BY DESC(?count)
```

Listing 3: The number of mashup categories the Google Maps API was used.

```
SELECT ?protocol (COUNT(?api) as ?count)
WHERE {
  ?api rdf:type lso:WebAPI .
  ?api prov:generatedAtTime ?date .
  ?api lso:supportedProtocol ?protocol .
  FILTER (?date >= "2013-01-01"^^xsd:dateTime
    && ?date < "2014-01-01"^^xsd:dateTime)
} ORDER BY DESC(?count)
LIMIT 5
```

Listing 4: The most popular API protocols in 2013.

Further, a Web API analyst might be interested in the latest trends in the API ecosystem. Questions such as *"What protocols and formats are supported the most by the APIs?"* or *"Which domains provided the most APIs in 2013?"* are likely to occur. Using the SPARQL query in Listing 4 we can get the top 5 most popular protocols in 2013, which is also illustrated in Figure 3 for the two most used protocols REST and SOAP, for a period of ten years.

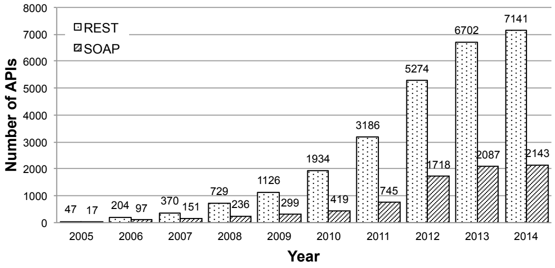

Fig. 3. Popularity of REST and SOAP protocols over time.

An answer to the question *"Which domains provided most APIs in 2013?"* can be answered with the SPARQL query in Listing 5.

```
SELECT ?category (COUNT(?api) as ?count)
WHERE {
  ?api rdf:type lso:WebAPI .
  ?api prov:generatedAtTime ?date .
  ?api lso:assignedCategory ?category .
  FILTER (?date > "2012-01-01"^^xsd:dateTime
    && ?date < "2013-01-01"^^xsd:dateTime)
} ORDER BY DESC(?count)
LIMIT 5
```

Listing 5: The most popular API categories in 2013.

The results show that the most popular API category is "tools", followed by the "science", "internet", "enterprise" and "financial" categories. It is interesting that the "financial" and the "enterprise" categories are among the top five most popular API categories, which indicates that APIs are already understood as a relevant technology also by other domains than the internet and the social networks.

A more in-depth analysis using the Linked Web APIs dataset has been conducted in [8]. In particular, the dataset has been used as a reference dataset for link discovery in RDF graphs.

7.2. *Survey on the Usefulness and Potential of the Dataset*

In order to evaluate the potential and the usefulness of the dataset we have executed a survey. The survey targeted people who consume, develop and/or provide Web APIs. In the survey participated 29 people and all of the participants stated that they have searched or used an API, while 19 stated that they also provide an API. The results from the survey show that most of the developers have difficulties while searching an API - 3.4% find it very hard, 41.4% hard, and 27.6% somewhat hard. In addition, the majority of the participants welcome a central API repository - 34.5% find it very helpful, 37.9% helpful, 20.7% somewhat helpful, and 6.9% little helpful. In the survey, we have asked the participants to indicate the usefulness of the Linked Web APIs dataset from the perspective of a Web API consumer and provider. The results (cf. Figure 4) show that both, the consumers and the providers find the dataset useful. The results also show that the dataset appears to be more useful for the API consumers than for the API providers.

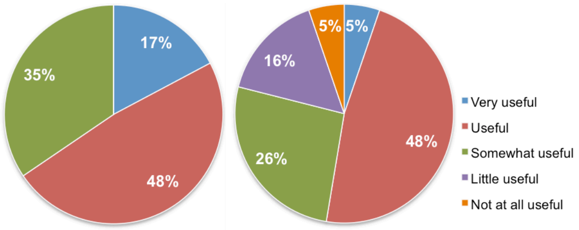

Fig. 4. Usefulness of the dataset as seen by the consumers (left) and providers (right).

From six participants who indicated that they have used ProgrammableWeb to search for an API, two indicated that they find the dataset useful, two useful and two somewhat useful.

We further break down the results on the usefulness of the dataset from perspective of users that search for APIs i) by using search engines such as Google, ii) running keyword-based search in service directories such as ProgrammableWeb, or iii) by asking other developers for an advice. The results are as follows:

- Using search engines such as Google (27 users); 18% very useful, 54% useful and 28% somewhat useful.
- Running keyword-based search in service directories such as ProgrammableWeb (6 users); 33% for very useful, useful and somewhat useful.
- Asking other developers for an advice (21 users); 19% very useful, 48% useful and 33% somewhat useful.

Further, in order to evaluate new potential third-party uses of the dataset, we have also asked the participants if they will consider using the dataset in the near future (cf. Figure 5). As shown in Figure 5, consumers have shown more interest in using the dataset than the providers.

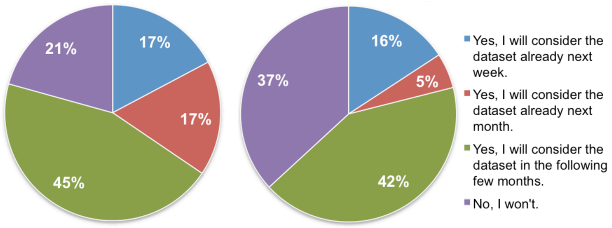

Fig. 5. Will consumers (left) and providers (rigth) consider the dataset in the near future.

Finally, in the survey we have evaluated the usefulness of the dataset on several use cases. The results are as follows:

- 86% – Find and select relevant APIs.
- 86% – Increase the visibility of the APIs.
- 62% – Evaluate the recent trends in the API ecosystem.
- 59% – Compare APIs to others.
- 52% – Automated composition of Web APIs.
- 38% – Track the popularity of the Web APIs.

It can be observed that our goals for the dataset are well aligned with the possible use cases, as seen by the Web API consumers and providers. The results from the survey are also available online[20].

[20] Results from the survey: `https://dx.doi.org/10.6084/m9.figshare.3459044.v2`

In overall, the results from the survey confirm the usefulness and the potential of the Linked Web APIs dataset.

8. Discussion and Future Work

8.1. Relation to Existing Ontologies

There are several proposals on machine readable descriptions for Web services. hRESTS [7], SADI [10], WSMO-lite [9], Hydra[21] and the Minimal Service Model (MSM)[22] ontology, define models for Semantic Web Service descriptions. The Linked Web APIs ontology builds on top of the WSMO-lite, hRESTS and the MSM Semantic Web Service models and in the near future we will also provide an alignment for the SADI model. SADI provides mechanism for an automated discovery, composition and orchestration of Web services. Since the Linked Web APIs dataset provides large amount of information about Web APIs, it can efficiently aid the process of discovery of relevant APIs for SADI composition workflows.

Moreover, there are also non-Semantic Web standards such as WADL[23] and WSDL[24], which define syntactic descriptions for Web services. These syntactic descriptions are of high importance for the process of execution of Web services, individually or combined in service compositions. APIs.json[25] is another API description format which has recently gained attention by the API community. It is a JSON based format for public deployment of API descriptions. In our future work, we plan to add support for these API description formats.

In our future work, we also plan to integrate ontologies such as the SPARQL Service Description[26] ontology and the DataID[27] dataset description model [2] which will in turn allow description of SPARQL processing services and corresponding Linked Data datasets. Last but not least we want to evaluate possible alignments of the ontology with tagging vocabularies such as the MUTO[28] and the SCOT[29] vocabularies.

8.2. Additional Data Sources

Currently, the Linked Web APIs dataset is populated with data from the ProgrammableWeb.com repository. Nevertheless, our ultimate goal is to establish the Linked Web APIs as a central Linked Data hub for Web API descriptions. In order to achieve this goal, we are currently working on enriching the dataset with API descriptions from other data sources. Integration of the API repository APIs.io as part of the Linked Web APIs dataset is a currently ongoing effort. The repository provides over 1,000 API descriptions in the APIs.json[30] format. The APIs.json descriptions are being deployed in a decentralized manner, at the same domain from which the APIs are available. By integrating the APIs.io repository as part of the Linked Web APIs dataset, developers are going to be able to publish and easily maintain their API descriptions, while at the same time, making theirs descriptions available as Linked Data. In our future work, we will also consider integrating API marketplaces, such as the *mashape* marketplace[31].

We also plan to enrich the dataset with user profiles from traditional social networks. We want to interlink the tags and categories information with relevant datasets from the LOD cloud such as the Wikidata[32], Wiktionary[33] and Dbnary[34]. Last but not least we want to explore other applications using the dataset and assess its potential.

9. Conclusion

The growing number of available Web APIs requires new mechanisms to support the process of sharing, discovery, integration and re-use of Web APIs at a large scale. In this paper, we have presented the Linked Web APIs dataset, the first Linked Data dataset which provides Web API descriptions. The dataset supports i) *API consumers*-in the process of discovery, selection and use of Web APIs, ii) *API providers*-in increasing the visibility and tracking the popularity of their Web APIs, and iii) *API analysts*-in analyzing the API ecosystem. The dataset will also help to raise the awareness about the importance of providing semantic

[21] http://www.hydra-cg.com/spec/latest/core/

[22] http://iserve.kmi.open.ac.uk/ns/msm#

[23] https://www.w3.org/Submission/wadl/

[24] https://www.w3.org/TR/wsdl20/

[25] http://apisjson.org/index.html

[26] http://www.w3.org/TR/sparql11-service-description/

[27] http://wiki.dbpedia.org/projects/dbpedia-dataid-unit

[28] http://muto.socialtagging.org/core/v1.html

[29] http://rdfs.org/scot/spec/

[30] http://apisjson.org/format.html

[31] https://market.mashape.com/

[32] https://www.wikidata.org

[33] https://www.wiktionary.org/

[34] http://kaiko.getalp.org/about-dbnary/

Web API descriptions and publishing them as Linked Data. The dataset has been validated in several recent works [4,5,8] in the context of personalized recommendations and link analysis. Also, on a set of usage scenarios we have shown the potential of the dataset.

Acknowledgement. We thank ProgrammableWeb.com for supporting this research.

References

[1] T. Berners-Lee. Linked data – design issues. 2006. Available from `https://www.w3.org/DesignIssues/LinkedData.html`.

[2] M. Brümmer, *et al.* DataID: Towards Semantically Rich Metadata for Complex Datasets. In *Proceedings of the 10th International Conference on Semantic Systems*, SEM '14, pp. 84–91. ACM, New York, NY, USA, 2014.

[3] M. Dojchinovski and T. Kliegr. Entityclassifier.eu: Real-time Classification of Entities in Text with Wikipedia. In H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, (eds.) *Machine Learning and Knowledge Discovery in Databases*, vol. 8190 of *Lecture Notes in Computer Science*, pp. 654–658. Springer Berlin Heidelberg, 2013.

[4] M. Dojchinovski, J. Kuchar, T. Vitvar, and M. Zaremba. Personalised Graph-Based Selection of Web APIs. In P. Cudré-Mauroux, *et al.*, (eds.) *The Semantic Web ISWC 2012*, vol. 7649 of *Lecture Notes in Computer Science*, pp. 34–48. Springer Berlin Heidelberg, 2012.

[5] M. Dojchinovski and T. Vitvar. Personalised Access to Linked Data. In K. Janowicz, S. Schlobach, P. Lambrix, and E. Hyvönen, (eds.) *Knowledge Engineering and Knowledge Management*, vol. 8876 of *Lecture Notes in Computer Science*, pp. 121–136. Springer International Publishing, 2014.

[6] K. Janowicz, *et al.* Five Stars of Linked Data Vocabulary Use. *Semantic Web*, 5(3):173–176, 2014.

[7] J. Kopecky, K. Gomadam, and T. Vitvar. hRESTS: An HTML Microformat for Describing RESTful Web Services. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, vol. 1, pp. 619–625. Dec 2008.

[8] J. Kuchar, M. Dojchinovski, and T. Vitvar. Time-aware Link Prediction in RDF Graphs. In *Knowledge Engineering and Knowledge Management*, vol. 8876 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014.

[9] T. Vitvar, J. Kopecký, J. Viskova, and D. Fensel. WSMO-Lite Annotations for Web Services. In S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, (eds.) *The Semantic Web: Research and Applications*, vol. 5021 of *Lecture Notes in Computer Science*, pp. 674–689. Springer Berlin Heidelberg, 2008.

[10] M. D. Wilkinson, B. Vandervalk, and L. McCarthy. SADI Semantic Web Services – 'cause you can't always GET what you want! In *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, pp. 13–18. IEEE, 2009.

[11] A. Zaveri, *et al.* Quality Assessment for Linked Data: A Survey. *Semantic Web Journal*, 7(1):63–93, 2016.