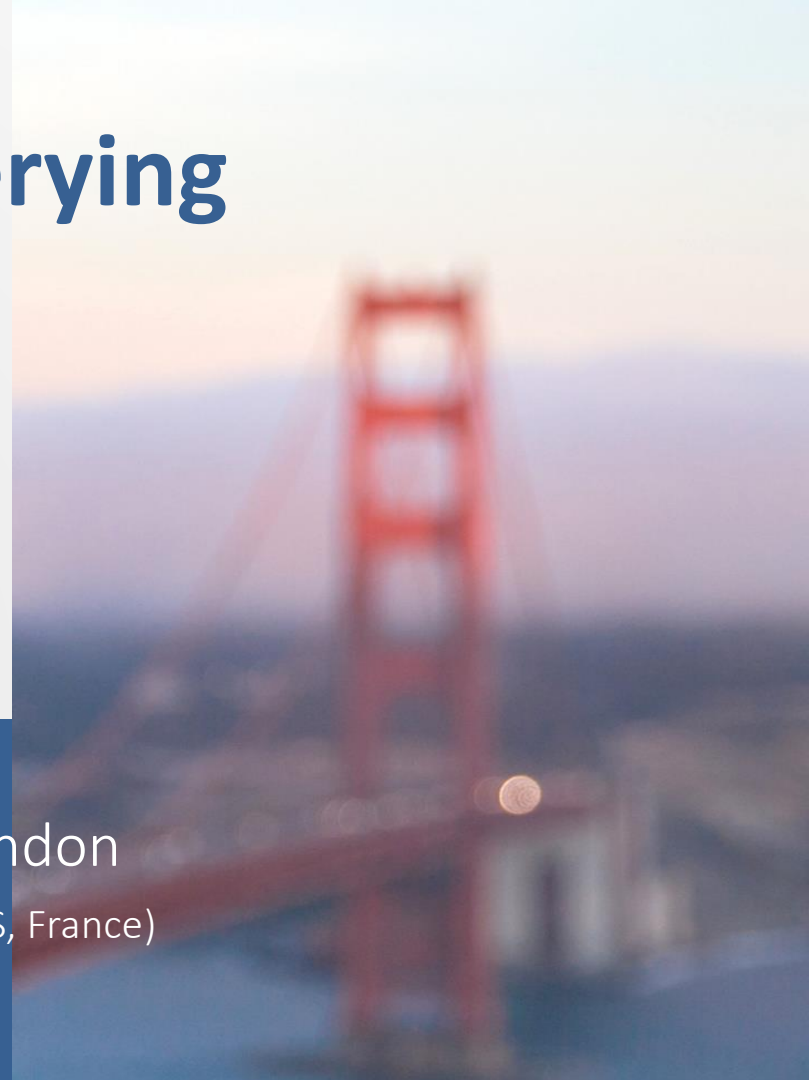
Enabling **Automatic Discovery** and **Querying** of **Web APIs** at **Web Scale** using **LD Standards**



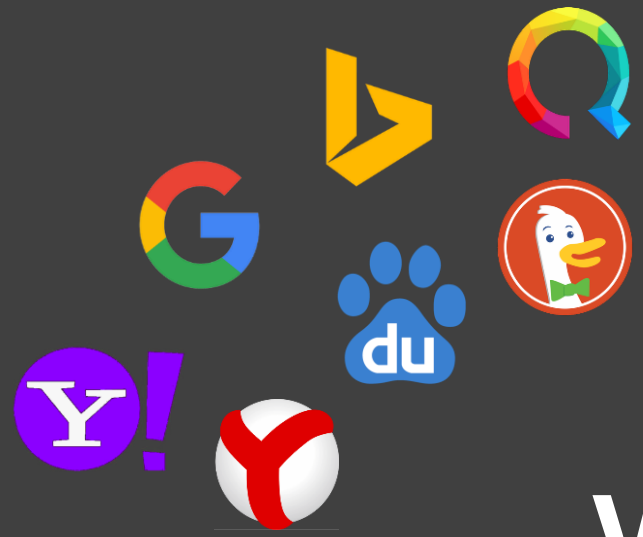
F. Michel, C. Faron-Zucker, O. Corby, F. Gandon

Wimmics* joint research team (Univ. Côte d'Azur, Inria, CNRS, I3S, France)

Linked Data on the Web and its Relationship with Distributed Ledgers (LDOW/LDDL)

13 May, 2019 - San Francisco, USA

* AI in bridging social semantics and formal semantics on the Web



CKAN

OpenDataSoft

datahub

Whom to ask to discover datasets?

ProgrammableWeb

LODStats

LODAtlas

voiD store

SPARQL Endpoints Status

No perfect discovery solution

Restricted scope (topic, format, API…)

Need to query multiple resources, accommodate disparate interfaces, mash up results

Manual dataset/service registration

Outdated metadata

Deprecated services

Limited relevance of results

Keyword-based: many irrelevant results.

Metadata-based: just a first step in the selection process.

No detailed insight into the data

What resources?

What properties?

What relationships?

3 principles to achieve **automatic discovery** and **consumption** of datasets at **Web scale**

1. Leverage Web search engines

Harvest data portals, exploit structured markup

2. Rich description of dataset/query services

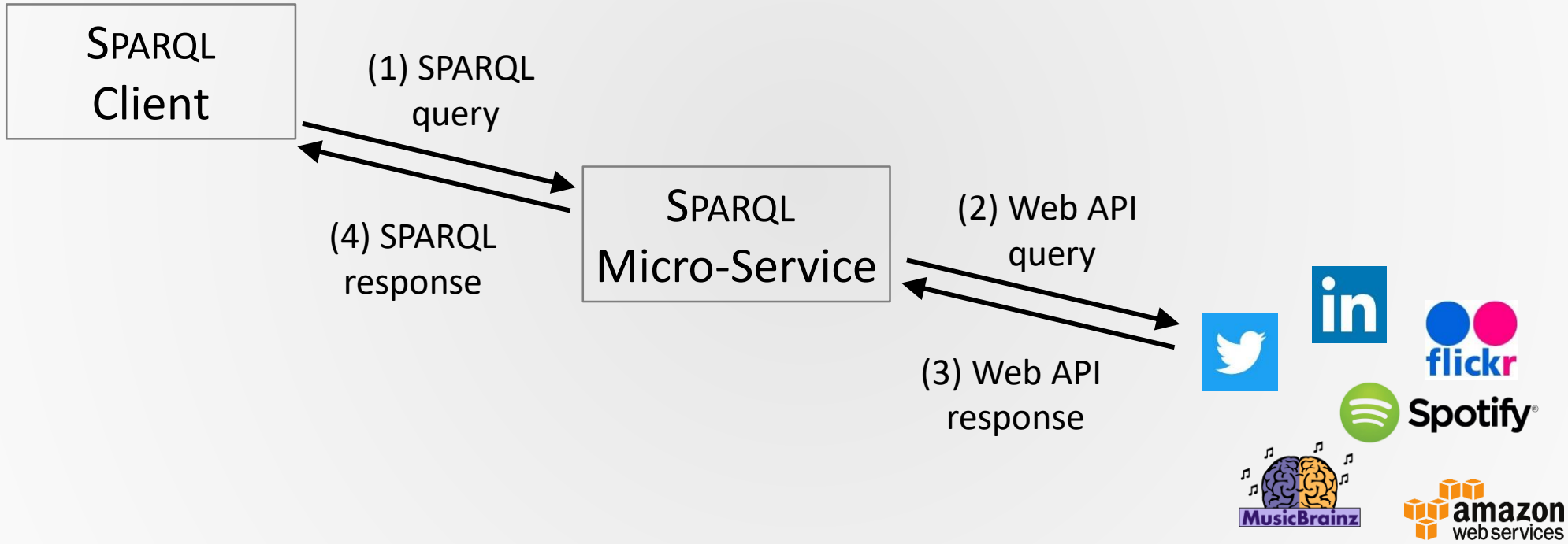
Machine-readable, beyond simple metadata

3. Rely on well-adopted (simple) standards

Need for consensus on technologies and practices

The SPARQL Micro-Service Architecture

Lightweight method to **query a Web API with SPARQL**, and assign dereferenceable URIs to Web API resources



*A SPARQL µ-service is a **CONFIGURABLE** SPARQL endpoint whose **ARGUMENTS** delineate the graph being queried.*

```
Endpoint: http://hostname/flickr/getPhotosByTag?tag=bridge

SELECT * WHERE {
    ?photo a schema:Photograph;
      schema:name       ?title;
      schema:contentUrl ?img.
}
```

Arguments passed as HTTP parameters

```
Endpoint: http://hostname/flickr/getPhotosByTag

SELECT * WHERE {
    ?photo a schema:Photograph;
      schema:keywords   "bridge";
      schema:name       ?title;
      schema:contentUrl ?img.
}
```

Arguments passed in the graph pattern

Example: search photos by tag using Flickr's Web API

```
SELECT * WHERE {
  ?photo a schema:Photograph;
    schema:keywords   "bridge";
    schema:name       ?title;
    schema:contentUrl ?img.
}
```

SPARQL μ-service

"bridge"

flickr

http://example.org/**flickr/getPhotosByTag**/

Machine-readable description of a dataset and its SPARQL micro-service

SPARQL **Service Description** as the **framework**

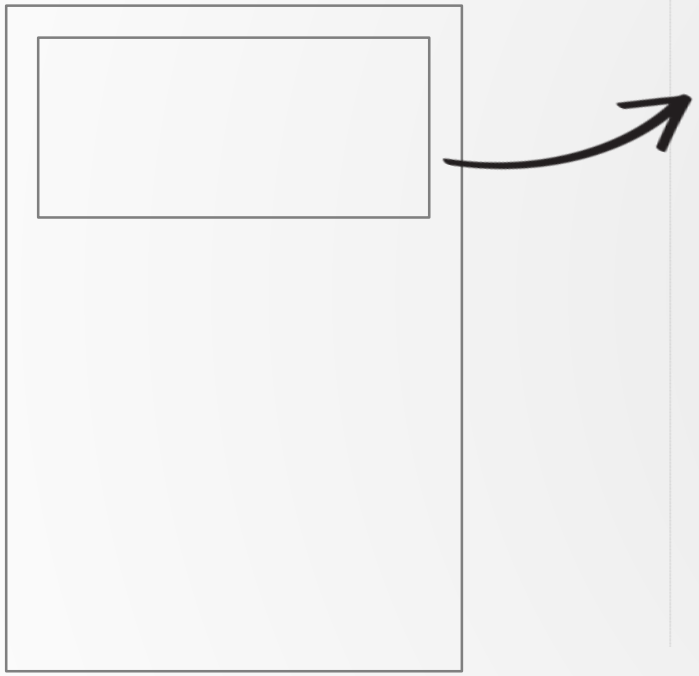
SPARQL SD document

http://example.org/flickr/getPhotosByTag/

Metadata about the dataset and micro-service

SPARQL SD document

http://example.org/flickr/getPhotosByTag/

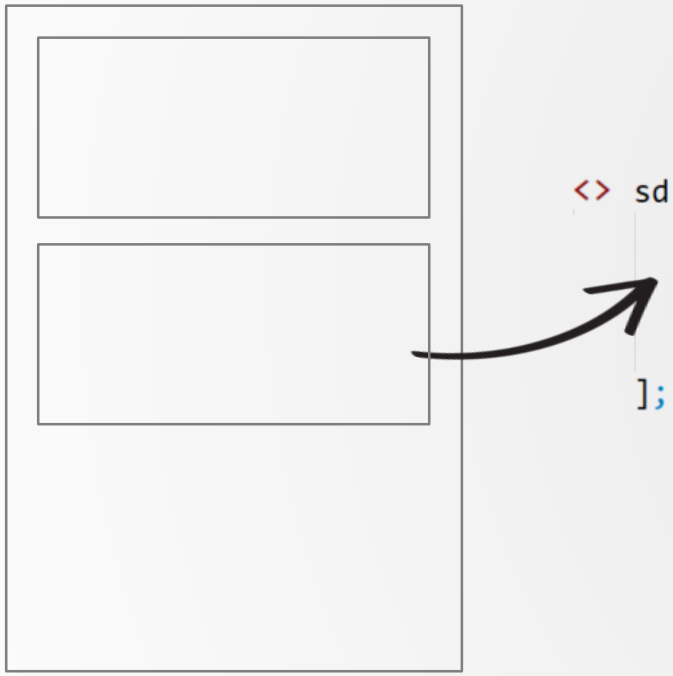


```
<>  a                    sd:Service;
    sd:endpoint          <>;
    sd:supportedLanguage sd:SPARQL11Query;
    sd:resultFormat
        <http://www.w3.org/ns/formats/SPARQL_Results_JSON>,
        <http://www.w3.org/ns/formats/Turtle> ... ;

    schema:name      "Search photos matching given tags";
    schema:keywords  "photography", "photo", "tag";
    schema:publisher [
        a schema:Organization;
        schema:name "University Cote d'Azur, CNRS, Inria";
        ...
    ];
    sms:exampleQuery '''
        prefix schema: <http://schema.org/>
        SELECT ?photo ?title ?img WHERE {
            ?photo a schema:Photograph;
                schema:keywords "brooklyn", "bicycle";
                schema:name ?title;
                schema:contentUrl ?img. }
        ''';
```

Specification of the graphs produced by the µ-service

SPARQL SD document
http://example.org/flickr/getPhotosByTag/



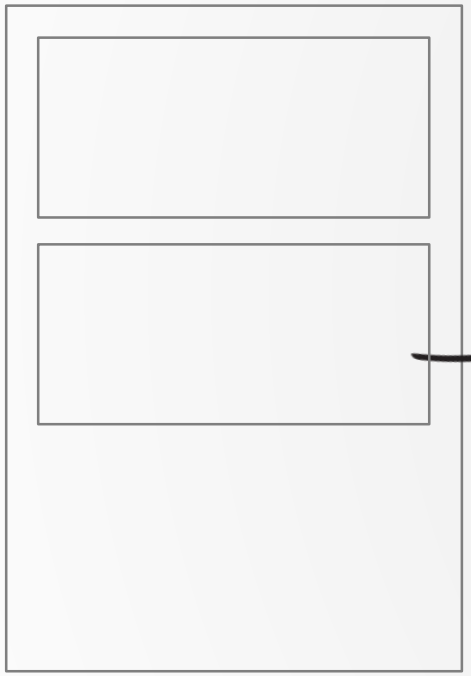
Goal: give insight into the data, so applications can decide whether the service is relevant for their goal

```
<> sd:defaultDataset [
       a                 sd:Dataset;
       sd:defaultGraph [ a sd:Graph; shacl:shapesGraph <ShapesGraph> ];
       sd:namedGraph   [ a sd:Graph; sd:name <ServiceDescription> ];
       sd:namedGraph   [ a sd:Graph; sd:name <ShapesGraph> ];
   ];
```

Specification of the graphs produced by the µ-service

Goal: give insight into the data, so applications can decide whether the service is relevant for their goal

SPARQL SD document
http://example.org/flickr/getPhotosByTag/



```
<> sd:defaultDataset [
       a                 sd:Dataset;
       sd:defaultGraph [ a sd:Graph; shacl:shapesGraph <ShapesGraph> ];
       sd:namedGraph   [ a sd:Graph; sd:name <ServiceDescription> ];
       sd:namedGraph   [ a sd:Graph; sd:name <ShapesGraph> ];
   ];
```

http://example.org/flickr/getPhotosByTag/ShapesGraph

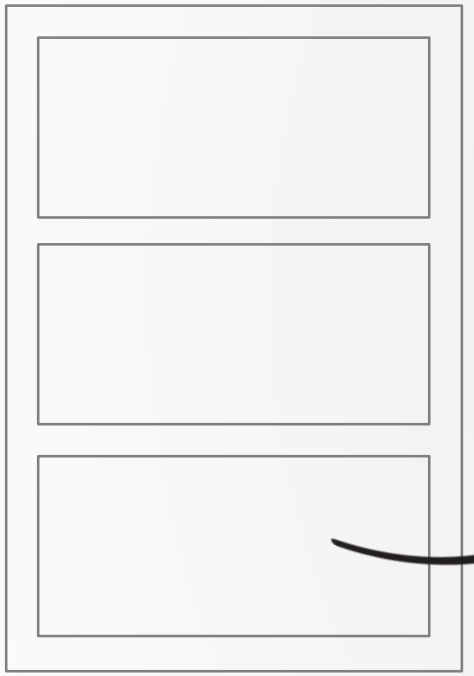
```
<ShapesGraph#PhotographShape> a shacl:NodeShape;
  shacl:property
    [ shacl:path rdf:type;          shacl:hasValue schema:Photograph ],
    [ shacl:path schema:name;       shacl:nodeKind shacl:Literal; ],
    [ shacl:path schema:contentUrl; shacl:nodeKind shacl:IRI ],
    <ShapesGraph#KwPropertyShape>;
  ...

<ShapesGraph#KwPropertyShape>
  a              shacl:PropertyShape;
  shacl:path     schema:keywords;
  shacl:nodeKind shacl:Literal;
  shacl:minCount 1.
```

Description of the **data source** and µ-service **arguments**

SPARQL SD document

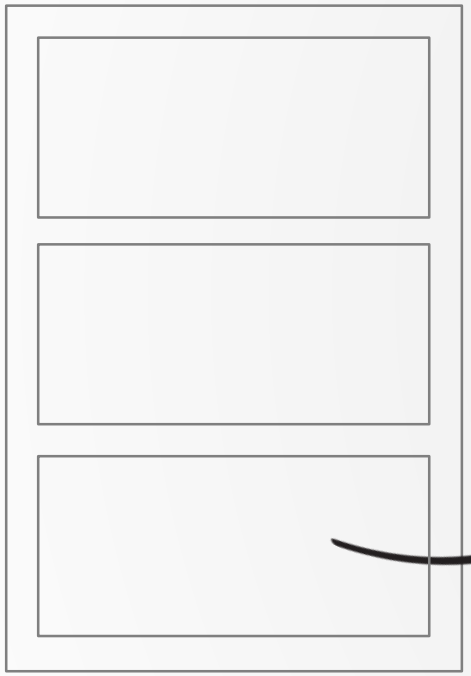
http://example.org/flickr/getPhotosByTag/



```
<> dct:source [
        a           schema:WebAPI;
        schema:name "Flickr API";
        schema:url  <https://www.flickr.com/services/api/>;
        schema:potentialAction [
            a schema:SearchAction, hydra:IriTemplate;
            hydra:template "https://api.flickr.com/?...&tags={tags}".
            hydra:mapping [
                hydra:variable "tags";
                hydra:required "true"^^xsd:boolean;
                hydra:property schema:keywords;
    ]]].
```

Description of the **data source** and µ-service **arguments**

SPARQL SD document
http://example.org/flickr/getPhotosByTag/



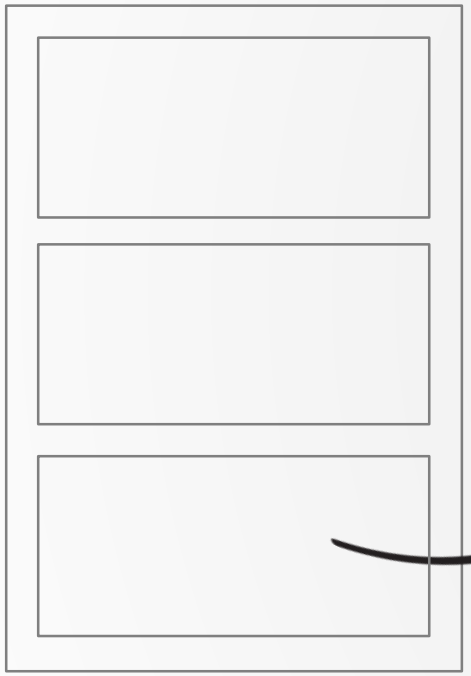
```
<> dct:source [
        a              schema:WebAPI;
        schema:name    "Flickr API";
        schema:url     <https://www.flickr.com/services/api/>;
        schema:potentialAction [
            a schema:SearchAction, hydra:IriTemplate;
            hydra:template "https://api.flickr.com/?...&tags={tags}".
            hydra:mapping [
                hydra:variable "tags";
                hydra:required "true"^^xsd:boolean;
                hydra:property schema:keywords;
    ]]].
```

```
SELECT * WHERE {
  ?photo a schema:Photograph;
    schema:keywords   "bridge";
    schema:name       ?title;
    schema:contentUrl ?img.
}
```

Description of the **data source** and µ-service **arguments**

SPARQL SD document

http://example.org/flickr/getPhotosByTag/



```
<> dct:source [
        a           schema:WebAPI;
        schema:name "Flickr API";
        schema:url  <https://www.flickr.com/services/api/>;
        schema:potentialAction [
            a schema:SearchAction, hydra:IriTemplate;
            hydra:template "https://api.flickr.com/?...&tags={tags}".
            hydra:mapping [
                hydra:variable "tags";
                hydra:required "true"^^xsd:boolean;
                shacl:sourceShape <ShapesGraph#KwPropertyShape>;
    ]]].
```

```
<ShapesGraph#PhotographShape> a shacl:NodeShape;
  shacl:property
    [ shacl:path rdf:type;          shacl:hasValue schema:Photograph ],
    [ shacl:path schema:name;       shacl:nodeKind shacl:Literal; ],
    [ shacl:path schema:contentUrl; shacl:nodeKind shacl:IRI ],
    <ShapesGraph#KwPropertyShape>;
  ...

<ShapesGraph#KwPropertyShape>
  a              shacl:PropertyShape;
  shacl:path     schema:keywords;
  shacl:nodeKind shacl:Literal;
  shacl:minCount 1.
```

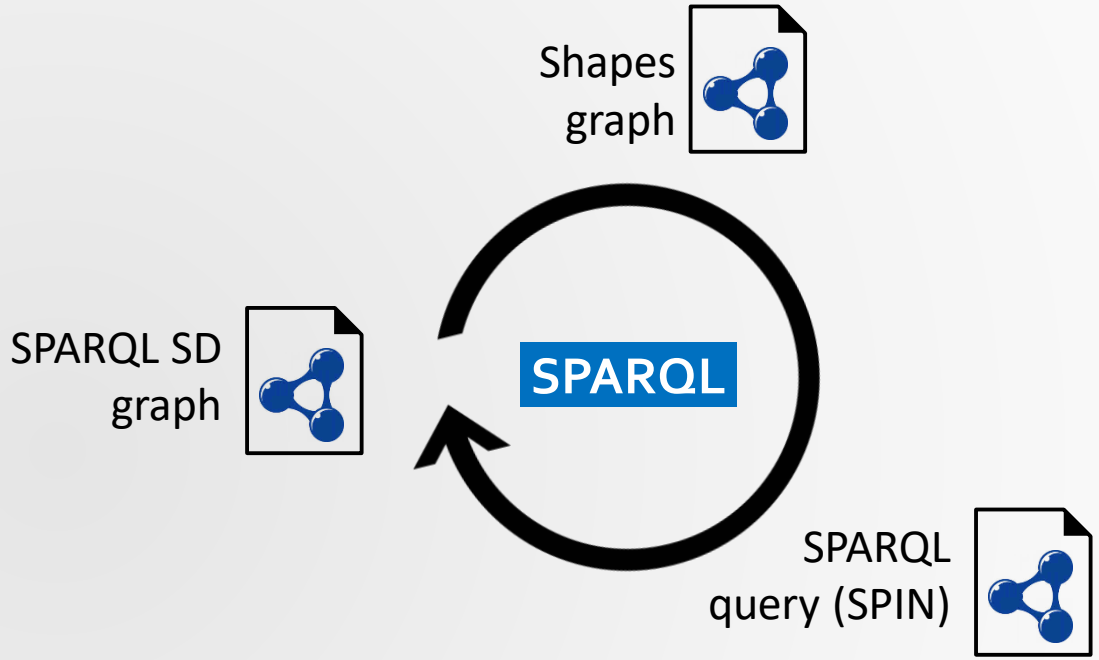
```
SELECT * WHERE {
  ?photo a schema:Photograph;
    schema:keywords   "bridge";
    schema:name       ?title;
    schema:contentUrl ?img.
}
```

SPARQL micro-service invocation

A **single** SPARQL query reasons upon

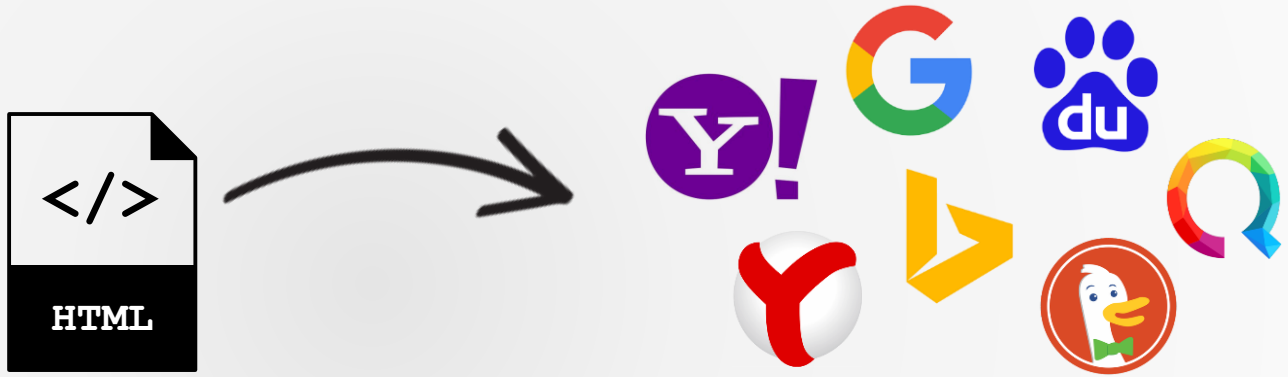
- the Service Description graph,
- the Shapes graph,
- the input query,

to extract the µ-service arguments

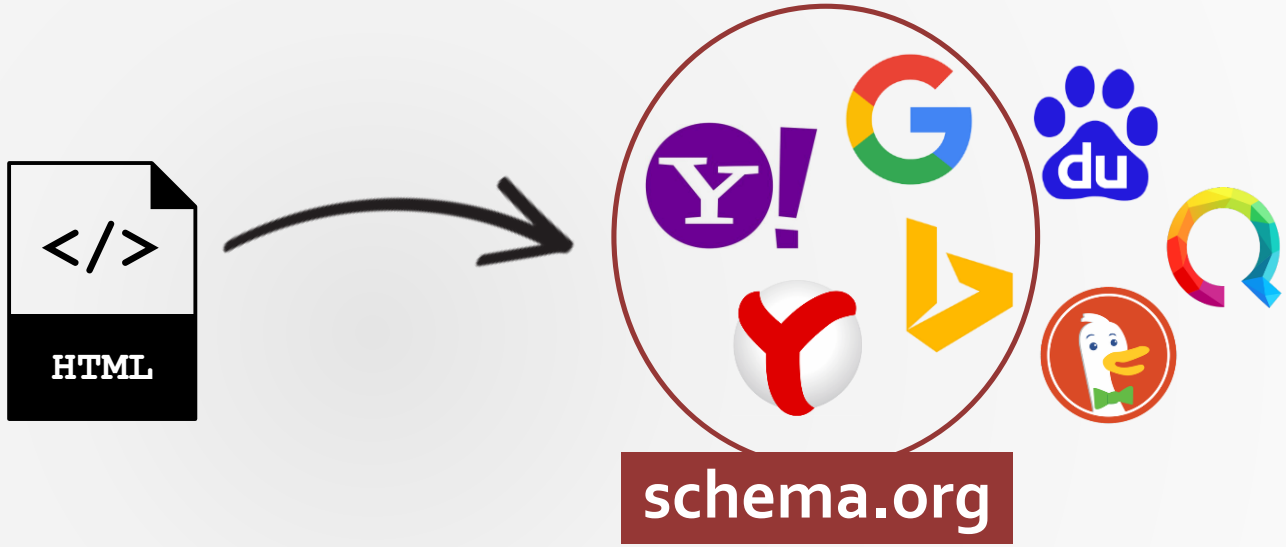


Discover a SPARQL micro-service using web search engines

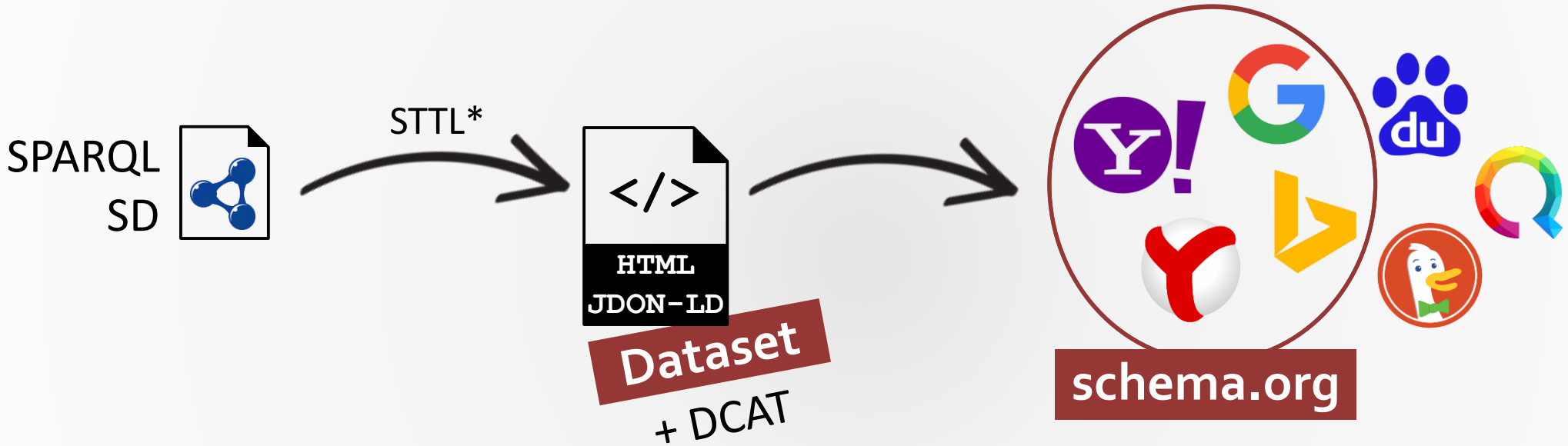
Discovery using search engines requires a web page



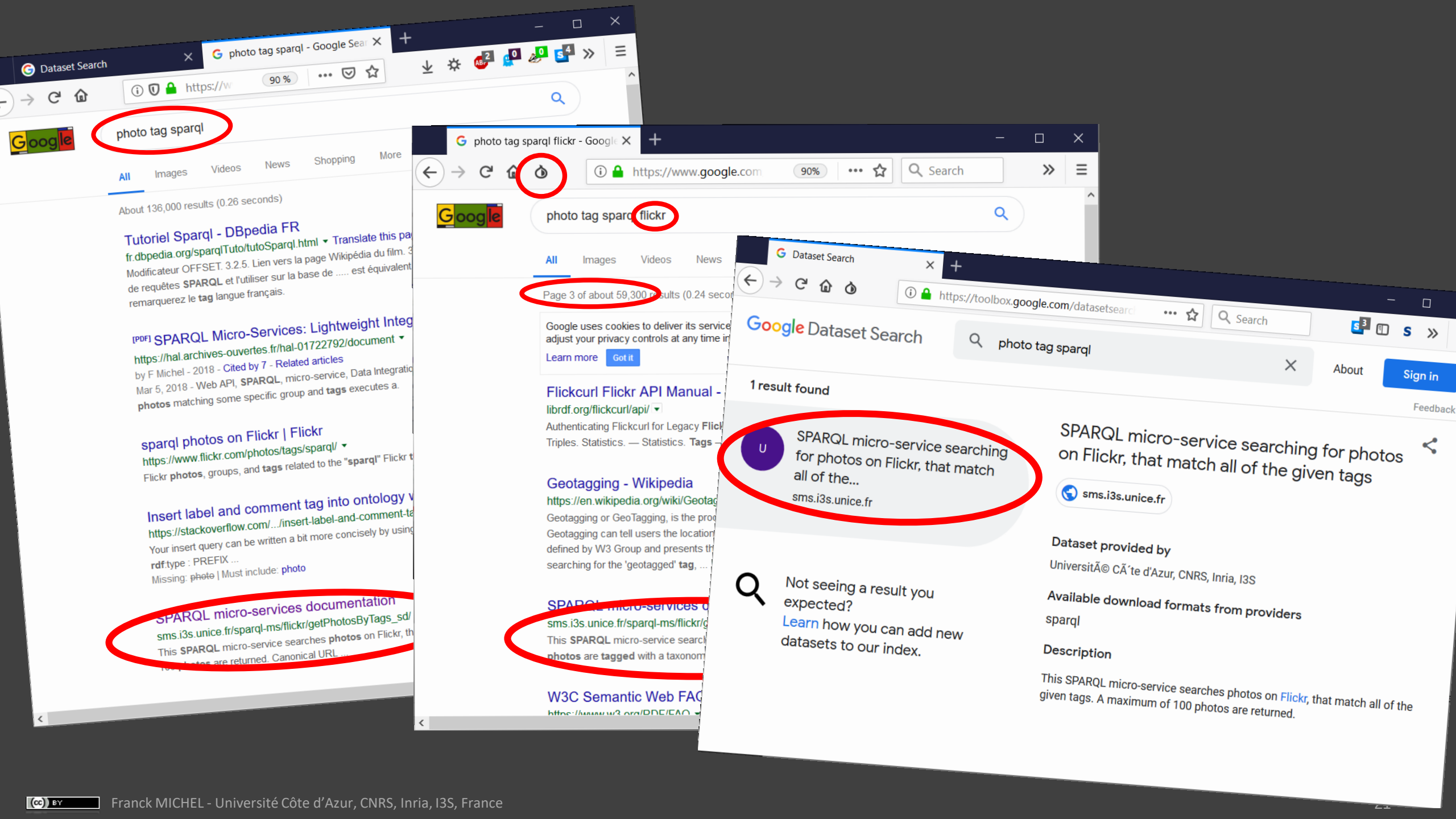
Discovery using search engines requires a web page



Web-scale discovery of SPARQL µ-services

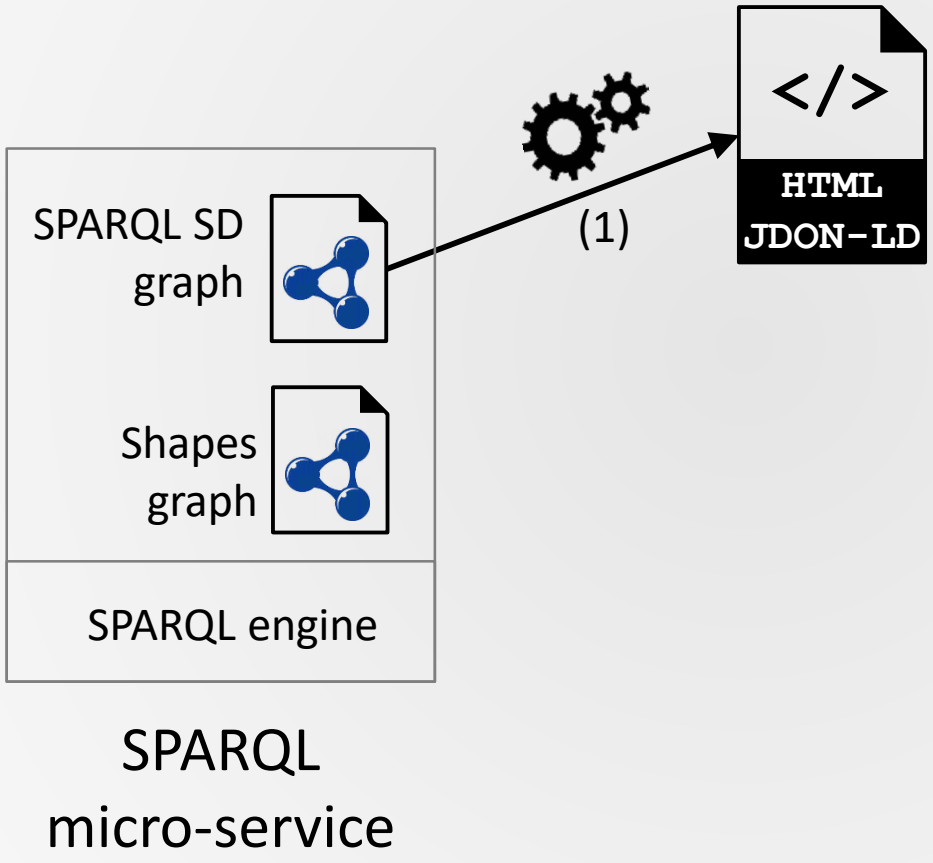


*STTL: SPARQL Template Transformation Language
http://ns.inria.fr/sparql-template/

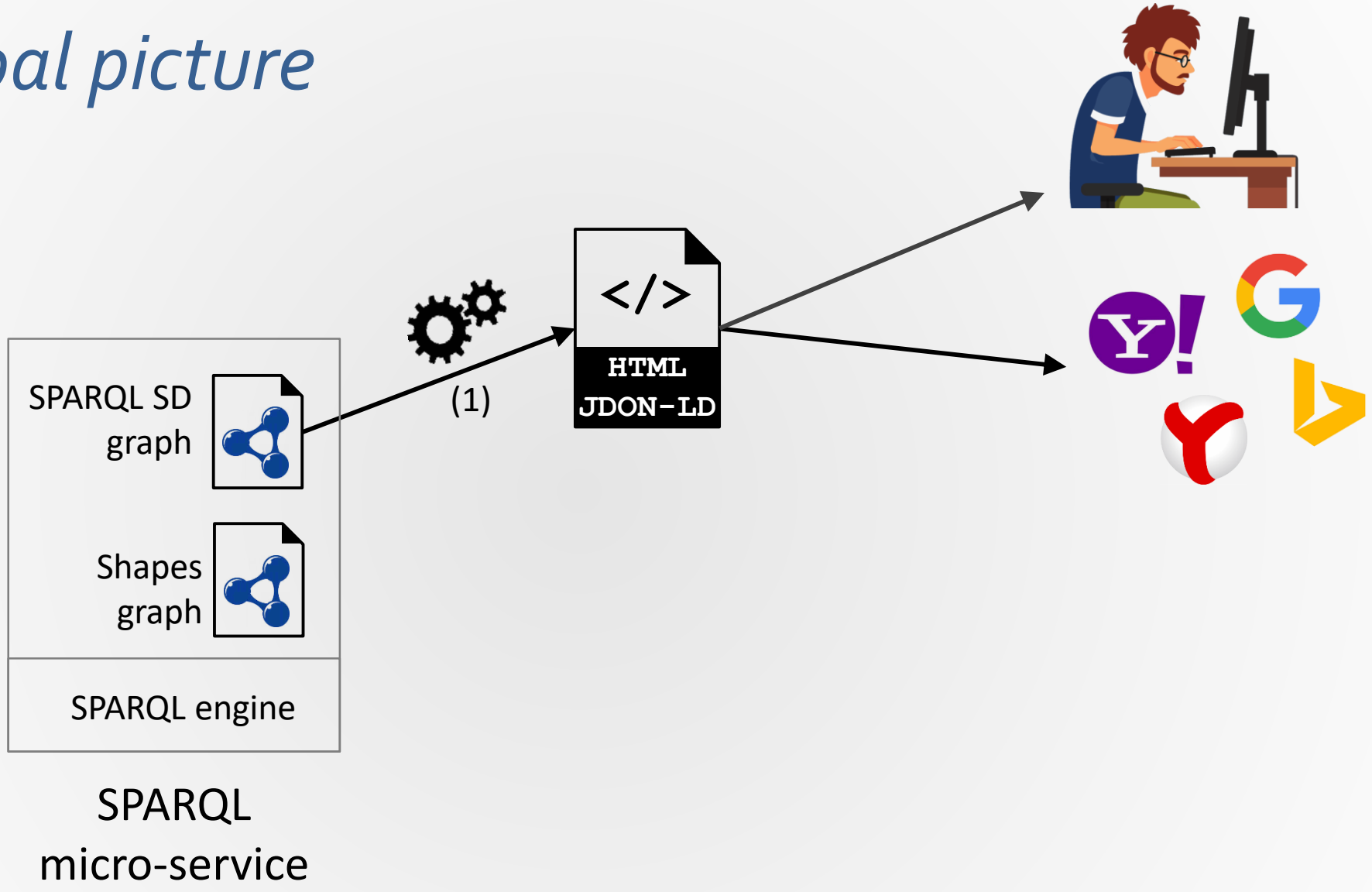

Dataset Search
photo tag sparql - Google Sea
https://w
90 %
photo tag sparql
Google
All
Images
Videos
News
Shopping
More
About 136,000 results (0.26 seconds)
Tutoriel Sparql - DBpedia FR
fr.dbpedia.org/sparqlTuto/tutoSparql.html
Translate this pa
Modificateur OFFSET. 3.2.5. Lien vers la page Wikipédia du film. 3
de requêtes SPARQL et l'utiliser sur la base de ..... est équivalent
remarquerez le tag langue français.
[PDF] SPARQL Micro-Services: Lightweight Integ
https://hal.archives-ouvertes.fr/hal-01722792/document
by F Michel - 2018 - Cited by 7 - Related articles
Mar 5, 2018 - Web API, SPARQL, micro-service, Data Integratio
photos matching some specific group and tags executes a.
sparql photos on Flickr | Flickr
https://www.flickr.com/photos/tags/sparql/
Flickr photos, groups, and tags related to the "sparql" Flickr t
Insert label and comment tag into ontology
https://stackoverflow.com/.../insert-label-and-comment-ta
Your insert query can be written a bit more concisely by using
rdf:type : PREFIX ...
Missing: photo | Must include: photo
SPARQL micro-services documentation
sms.i3s.unice.fr/sparql-ms/flickr/getPhotosByTags_sd/
This SPARQL micro-service searches photos on Flickr, th
are returned. Canonical URL
photo tag sparql flickr - Google
https://www.google.com
90%
Search
photo tag sparql flickr
Google
All
Images
Videos
News
Page 3 of about 59,300 results (0.24 secon
Google uses cookies to deliver its service
adjust your privacy controls at any time in
Learn more
Got it
Flickcurl Flickr API Manual -
librdf.org/flickcurl/api/
Authenticating Flickcurl for Legacy Flick
Triples. Statistics. — Statistics. Tags
Geotagging - Wikipedia
https://en.wikipedia.org/wiki/Geotag
Geotagging or GeoTagging, is the proc
Geotagging can tell users the location
defined by W3 Group and presents th
searching for the 'geotagged' tag, ...
SPARQL micro-services d
sms.i3s.unice.fr/sparql-ms/flickr/g
This SPARQL micro-service search
photos are tagged with a taxonom
W3C Semantic Web FAQ
https://www.w3.org/RDF/FAQ
Dataset Search
https://toolbox.google.com/datasetsearc
Search
Google Dataset Search
photo tag sparql
About
Sign in
Feedback
1 result found
U
SPARQL micro-service searching for photos on Flickr, that match all of the...
sms.i3s.unice.fr
Not seeing a result you expected?
Learn how you can add new datasets to our index.
SPARQL micro-service searching for photos on Flickr, that match all of the given tags
sms.i3s.unice.fr
Dataset provided by
UniversitÃ© CÃ´te d'Azur, CNRS, Inria, I3S
Available download formats from providers
sparql
Description
This SPARQL micro-service searches photos on Flickr, that match all of the given tags. A maximum of 100 photos are returned.

< WRAP-UP >

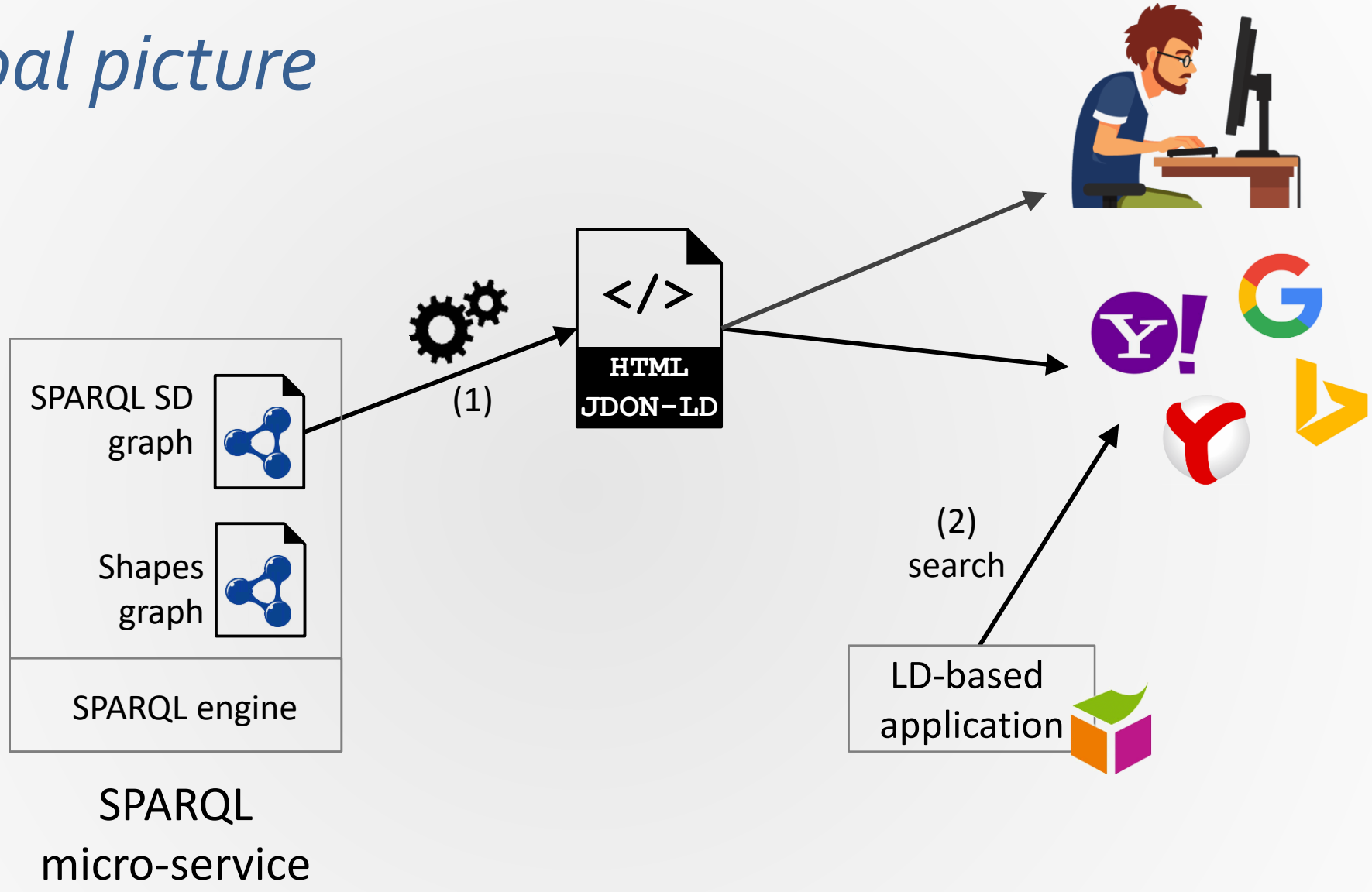
The global picture



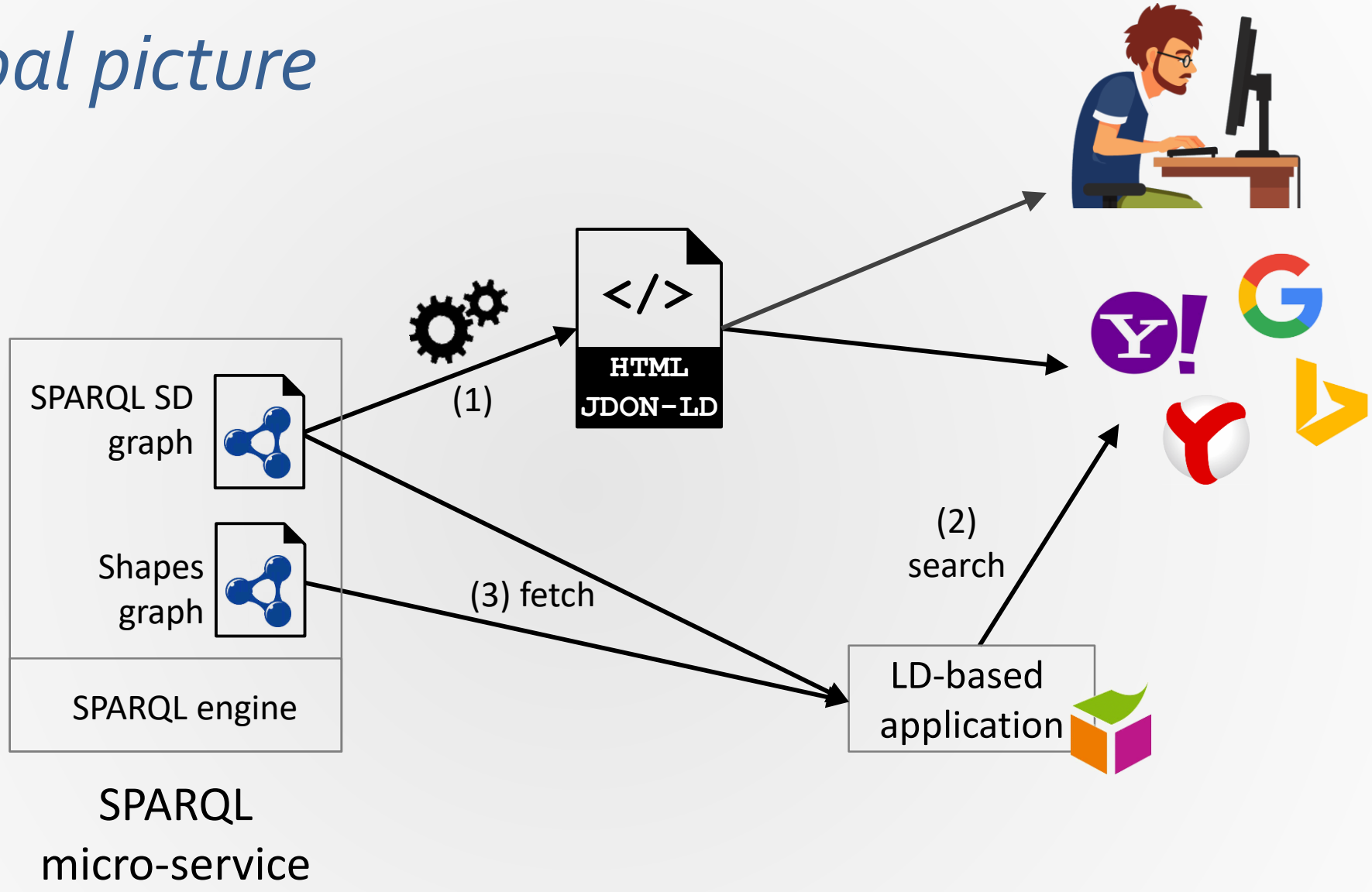
The global picture



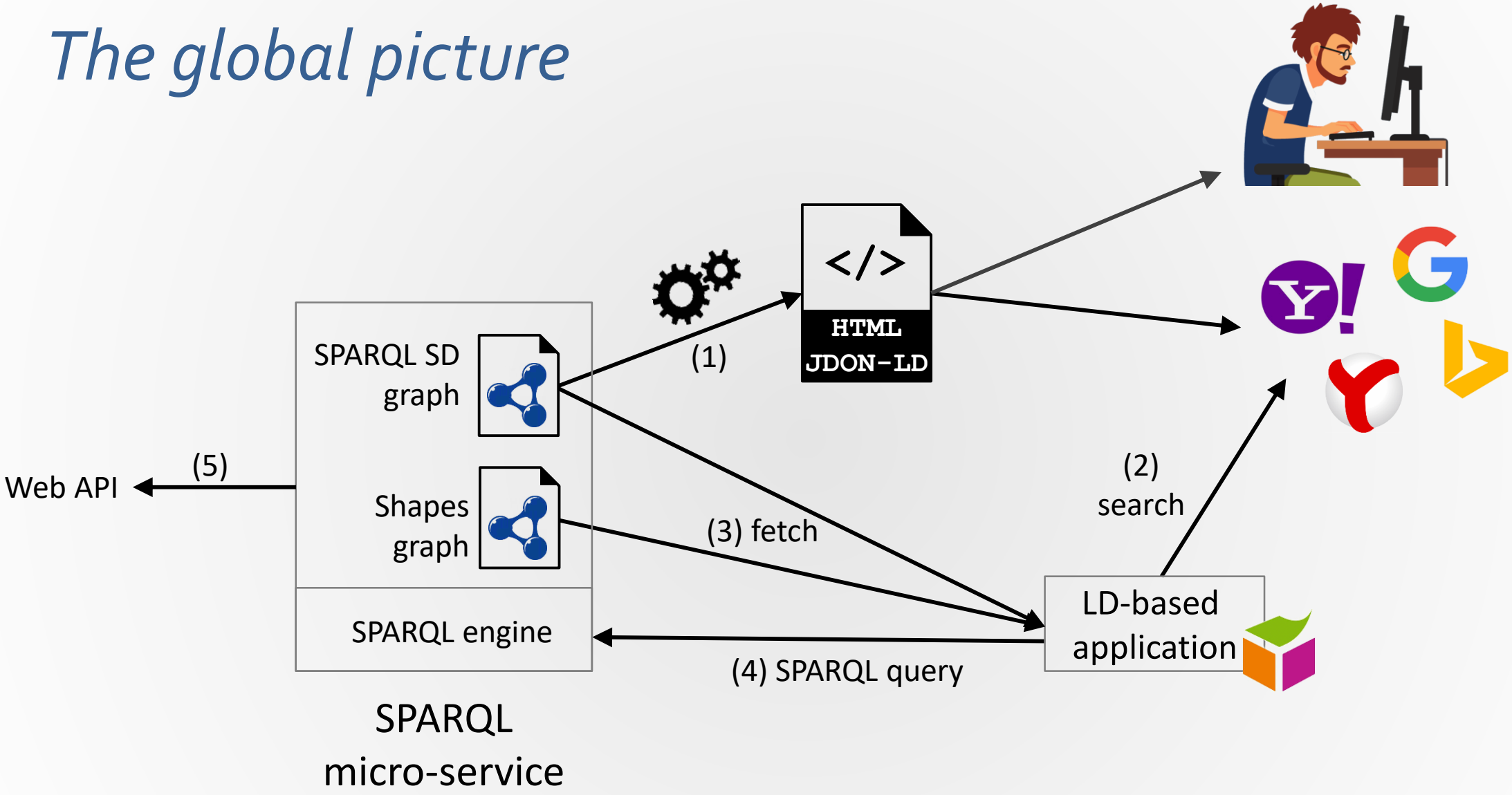
The global picture



The global picture



The global picture



3 principles to achieve **automatic discovery** and **consumption** of SPARQL micro-services at **Web scale**

1. Leverage Web search engines

Machine-readable description to web page + Schema.org/DCAT markup data

2. Rich description of dataset/query services

Metadata, SHACL, Schema.org, Hydra

3. Rely on well-adopted (simple) standards

SPARQL SD, SHACL, Schema.org/DCAT


Perspectives
3636
HONDA
ODYSSEY
TOYOTA

Perspectives

Demonstrate the whole chain

An application seeks to answer a query / achieve a goal:

- Discovers candidate services using search engines
- Selects relevant services based on SD/Shapes graphs
- Computes & enacts valid compositions

Extend SPARQL federated query engines

- Reason on SPARQL µ-services descriptions
- Query plan respecting services' inputs

Perspectives

Schema.org unpractical to denote dataset interfaces (API, endpoint…)

- WebAPI type extensions (EntryPoint)
- Convergence with DCAT 1.2 DataService

Google Dataset Search more effective than generic web search engines: more dataset search engines in the future?

Three principles,
many potential architectural/modelling choices
in different contexts

Thank-you

Citation:
F. Michel, C. Faron-Zucker, O. Corby & F. Gandon. **Enabling Automatic Discovery and Querying of Web APIs at Web Scale using Linked Data Standards**.
In Companion Proceedings of the 2019 World Wide Web Conference (WWW '19 Companion), 2019, San Francisco, CA, USA.

https://github.com/frmichel/sparql-micro-service

https://hub.docker.com/u/frmichel