Linked Data Notifications for RDF Streams

Jean-Paul Calbimonte

Institute of Information Systems

University of Applied Sciences and Arts Western Switzerland (HES-SO Valais-Wallis)

@jpcik

International Semantic Web Conference ISWC

Vienna, October 2017

HES-SO:
University of Applied Sciences and Arts Western Switzerland

We are here,
surrounded by mountains!

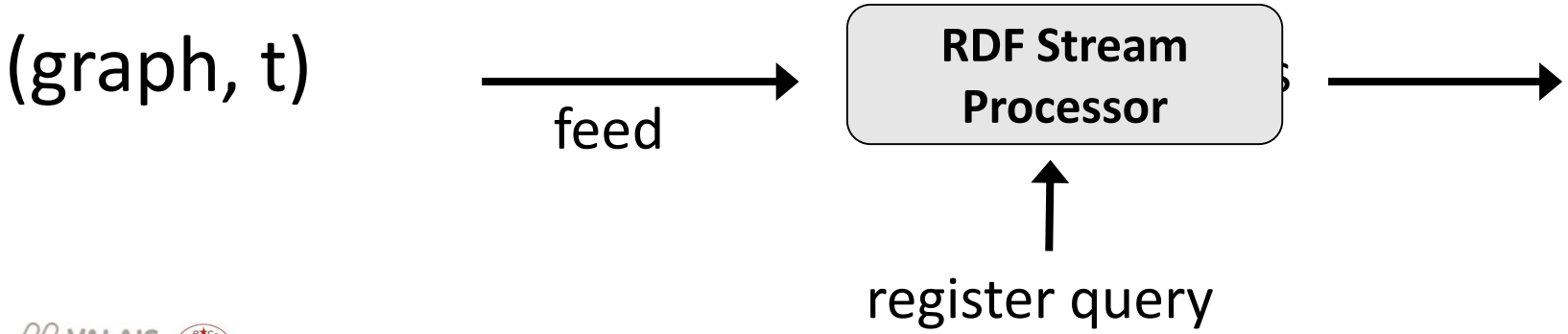
1

Linked Data Notifications for RDF Streams

RDF Streams

RDF graph: triples

RDF stream graph: graph+ timestamp



RSP Data Model

Timestamped Graph

```
:g1 {:axel :isIn :RedRoom. :darko :isIn :RedRoom}
{:g1 prov:generatedAtTime "2001-10-26T21:32:52"}
```

Allows:

- → **Many/One-triple graphs**
- → **Multiple time predicates**
- → **Implicit timestamp**
- → **Different timestamp representations**
- → **Contemporaneity**

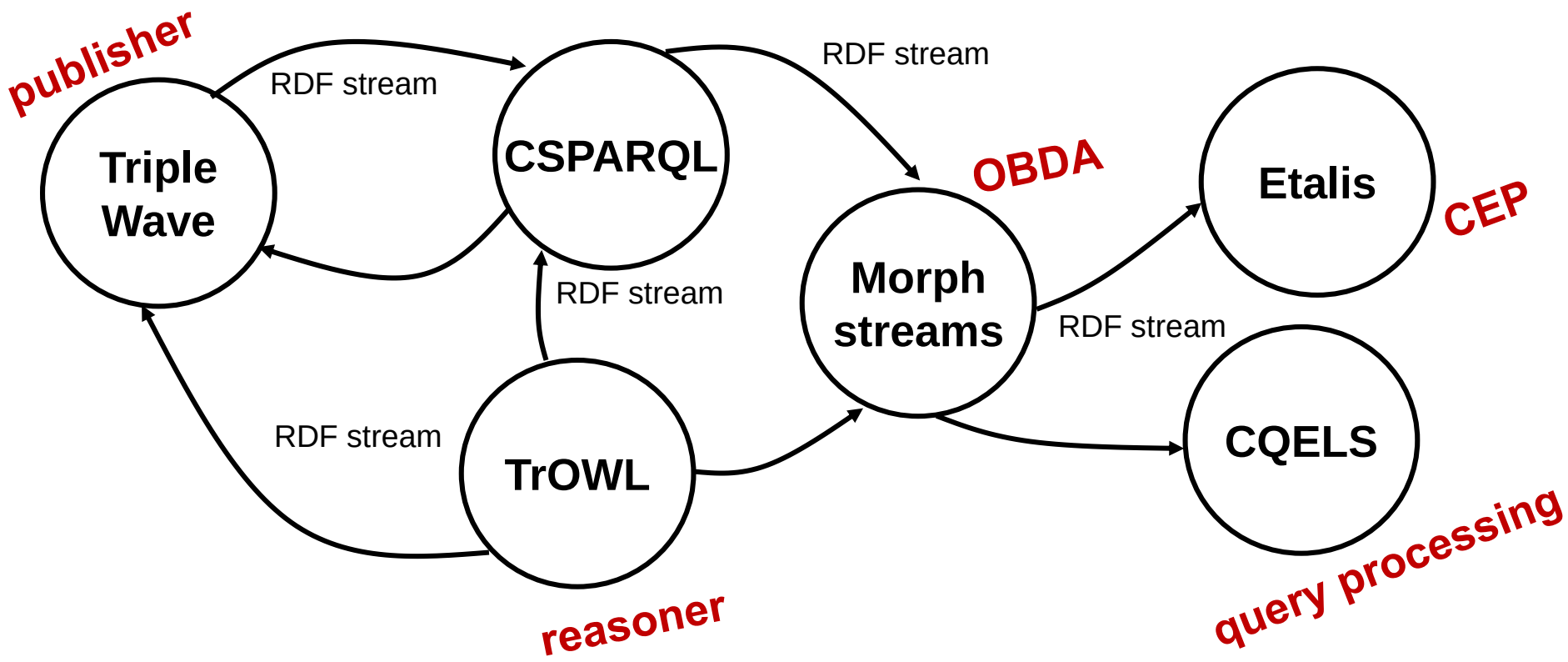
RDF Stream

A *RDF stream* S consists of a sequence of timestamped graphs (with a partial order)

```
:g1 {:axel :isIn :RedRoom. :darko :isIn :RedRoom} {:g1,prov:generatedAtTime,t1}
:g2 {:axel :isIn :BlueRoom. }                     {:g2,prov:generatedAtTime,t2}
:g3 {:minh :isIn :RedRoom. }                      {:g3,prov:generatedAtTime,t3}
...
```

https://www.w3.org/community/rsp/
http://w3id.org/rsp/abstract-syntax

RDF Stream Processors



2

Linked Data Notifications for RDF Streams

Linked Data Notifications

W3C Recommendation 2 May 2017

This version

https://www.w3.org/TR/2017/REC-ldn-20170502/

Latest published version

https://www.w3.org/TR/ldn/

Previous version:

https://www.w3.org/TR/2017/PR-ldn-20170321/

Latest editor's draft

https://linkedresearch.org/ldn/

Editors

Sarven Capadisli, University of Bonn, info@csarven.ca

Amy Guy, University of Edinburgh, amy@rhiaro.co.uk

LDN: Basics

Target

Resource which notifications is to/about

Sender

sends notifications

Consumer

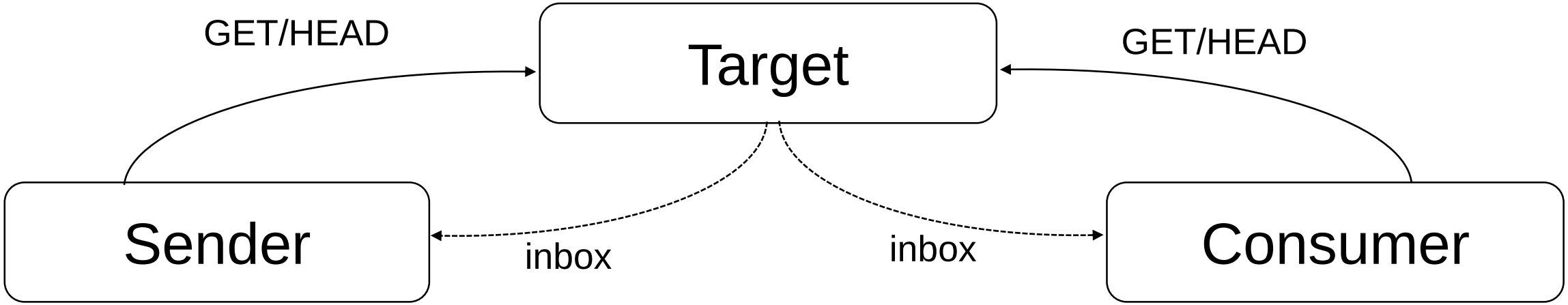
consumes notifications

Receiver

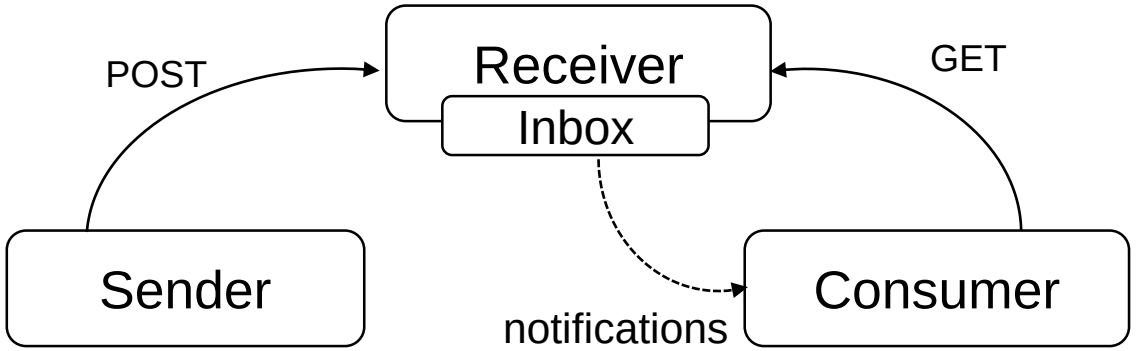
inbox

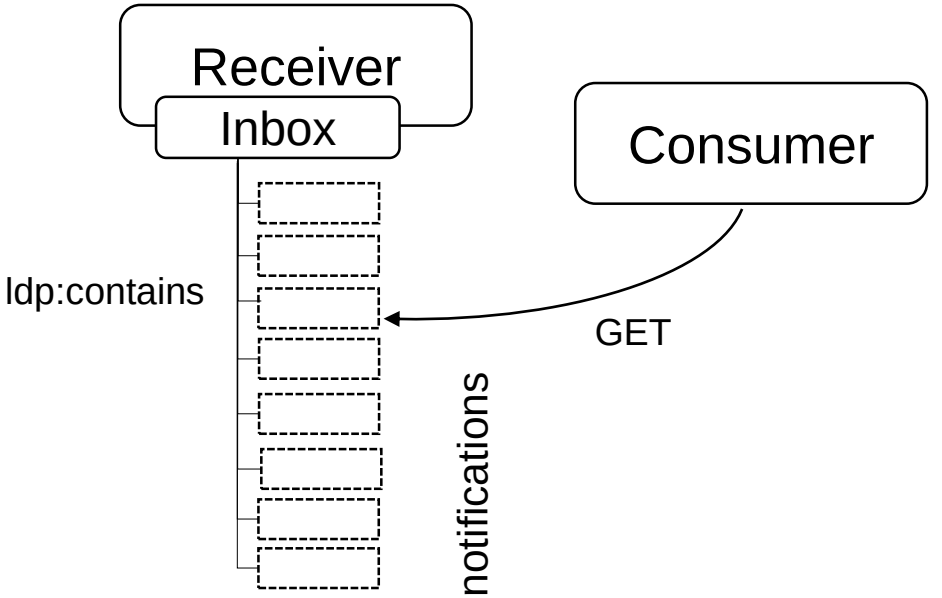
exposes notifications through inbox
creates notifications in inbox

LDN: Discovery



LDN Interactions

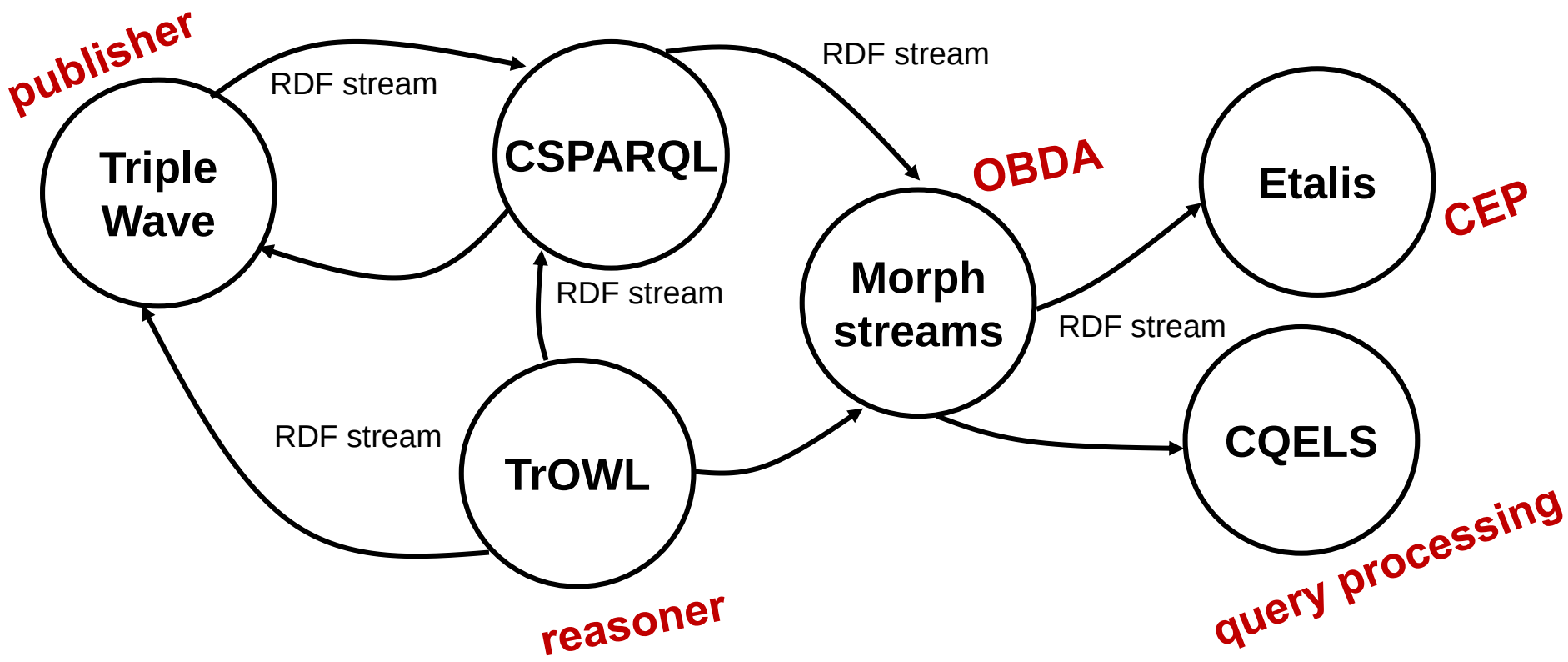




3

LDN for RDF Streams

We think LDN can help to get here:



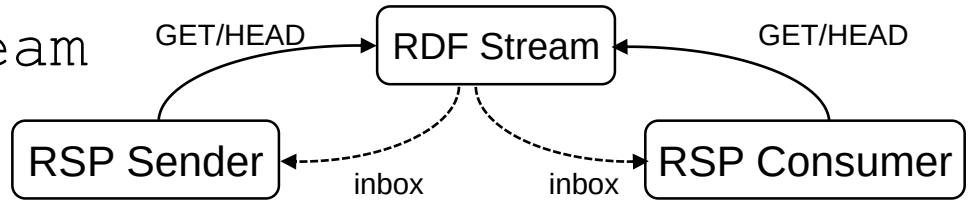
Streams and IRIs

- An RDF stream is uniquely identified by an IRI
- Stream IRI: obtain information about the stream
  - endpoints
- RDF stream is a read/write Web resource detached from potentially multiple endpoints used to interact with its contents.

Endpoint discovery

The endpoints of an RDF stream:

```
GET http://example.org/streams/my-stream
```



Response should include metadata about the stream:

```
{
  "@context": "http://www.w3.org/ns/ldp",
  "@id": "http://example.org/streams/my-stream",
  "inbox": "http://example.org/streams/my-stream/inbox"
}
```

Input/output stream

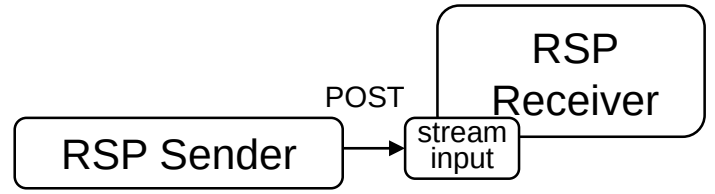
- Specialize it in two distinct types: an input inbox and an output inbox.
- Input stream: receiving notifications (i.e. to be fed) by senders.
- Output stream: only meant to be consumed, as they are produced by an RSP engine.

```
{
  "@context": "http://w3id.org/rsp/ldn-s",
  "@id": "http://example.org/streams/my-stream",
  "input": "http://example.org/streams/my-stream/input"
}
```

Sending stream notification

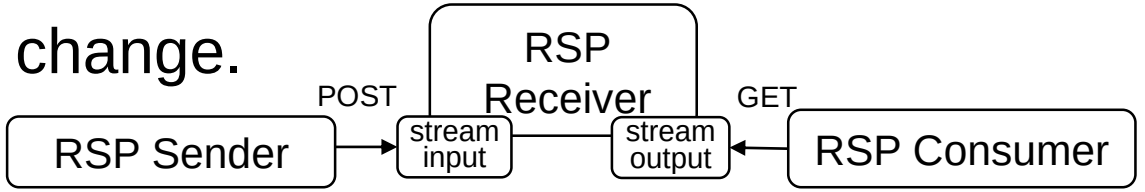
- `POST` stream elements
- body should contain the stream element that will be fed to the stream

```
POST /streams/my-stream/input HTTP/1.1
Host: example.org
Content-Type: application/ld+json
{
 "prov:generatedAtTime": "2017-07-22T05:00:00.000Z",
 "@id": "ex:Graph1",
 "@graph": [
   { "@id": "ex:humidityObservation",
     "ex:hasValue": 34.5}],
 "@context": {
   "prov": "http://www.w3.org/ns/prov#",
   "ex": "http://example.org#"} }
```



Publicizing stream elements

- `GET` stream elements from an RDF stream endpoint
- return the notification URIs listed as objects to the LDP `ldp:contains` predicate.
- stream elements "fade" with time
- listed stream contents may progressively change.




```
{
 "@context": "http://www.w3.org/ns/ldp",
 "@id": "http://example.org/streams/my-stream/output",
 "contains": [
   "http://example.org/streams/my-stream/output/graph1",
   "http://example.org/streams/my-stream/output/graph2" ]
}
```

Pulling stream elements

- Consumer explicitly requests for stream sub-sequences
- Practical to limit through size, time, filter parameters

```
{
  "@context": {
  "prov": "http://www.w3.org/ns/prov#",
  "ex": "http://example.org#"},
  "@graph": [
   { "prov:generatedAtTime": "2017-07-22T05:00:00.000Z",
     "@id": "ex:Graph1",
     "@graph": [
       { "@id": "ex:humidityObservation","ex:hasValue": 34.5 }]
   },
   { "prov:generatedAtTime": "2017-07-22T06:00:00.000Z",
     "@id": "ex:Graph2",
     "@graph": [
       { "@id": "ex:humidityObservation","ex:hasValue": 44.5 }]
   }
  ]
}
```

Pushing stream elements

Proactively send stream elements to the consumer

Example: Server-Sent Events protocol (HTTP-based).

- Continuously push RDF stream elements,
- One-directional (vs. bidirectional in WebSocket)
- Each data item is prefixed by the `data:` annotation.

Provide additional push protocols (different endpoints)
diverges from LDN → can only advertise one inbox

Register a query

- An actor may `POST` a query to an RSP endpoint
- Query must reference a valid registered RDF stream.
- RSP endpoint should return the URI of the resulting output stream, so that its results can be retrieved (pulling or pushing).

RDF stream:

```
http://example.org/streams/my-stream
```

Query:

```
SELECT ?s ?p ?o
WHERE {
   STREAM <http://example.org/streams/my-stream> [RANGE 2s]
   {?s ?p ?o}
}
```

LDN for RDF streams

- Simple, generic, extensible protocol
- Encapsulate behavior or heterogeneous implementations
- Use of existing Standards/recommendations
- Decentralized communication
- Potential for interoperability

On a Web of Data Streams

Daniele Dell'Aglio[1,a], Danh Le Phuoc[2,b], Anh Le-Tuan[3,c], Muhammad Intizar Ali[3,d], Jean-Paul Calbimonte[4,e]

http://w3id.org/wesp/web-of-data-streams

gracias! ¿tienes preguntas?

@jpcik

Jean-Paul Calbimonte

University of Applied Sciences and Arts Western Switzerland

HES-SO Valais-Wallis