Open APIs for Open Minds

QuantumLeap: Managing the Stream of Context Information History using Time Series DBs

Federico M. Facca

CTO, Martel Innovate

federico.facca@martel-innovate.com

FIWARE

Intro: What is QuantumLeap?

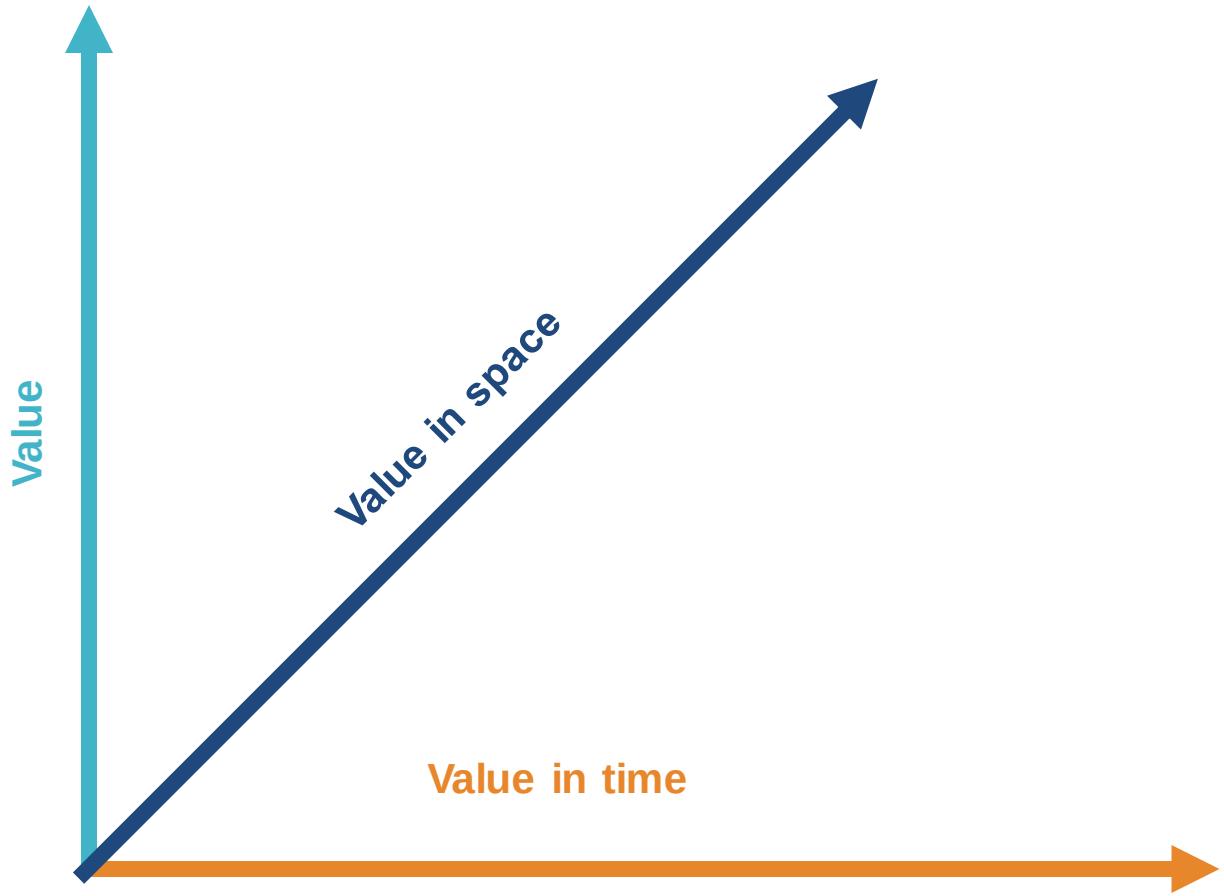
QuantumLeap in FIWARE

- **Core Context Management Chapter**
  - Cygnus
  - STH
  - **QuantumLeap**

- **Why QuantumLeap**
  1. Provide historical data support for NGSIv2 data
  2. Leverage on an efficient time-series database
  3. Deploy and scale it easily on containerized environments

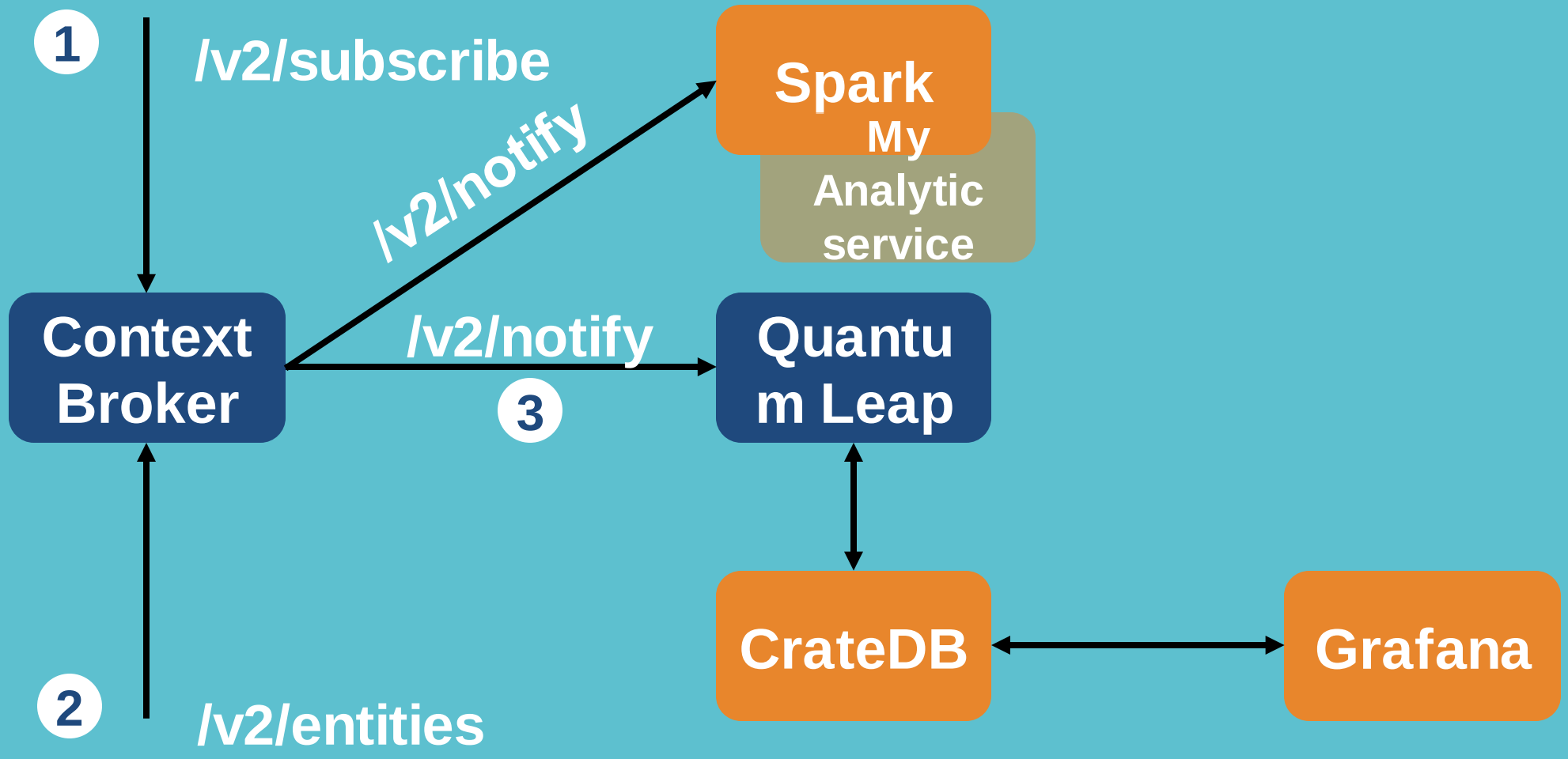
Data is no-more mono dimensional

- The attribute
  - Temperature is 20
- The history of the attribute
  - Temperature is 20 on 19th May 8:59:01.020 AM CEST
  - Temperature is 21 on 19th May 9:12:03.045 AM CEST
- The geo-localisation of the attribute
  - Temperature is 20 in 41°51'16.8"N 12°28'15.0"E



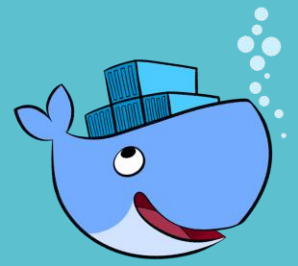
The core of our solution

Architecture Overview



QuantumLeap in 4 steps

Step 1: Deploy the services



- A) Simple deployment with docker-compose (for dev)
  - With the docker compose in QuantumLeap's repo:
    - https://github.com/smartsdk/ngsi-timeseries-api/blob/master/docker/docker-compose-dev.yml
  - Or as shown in the FIWARE step-by-step tutorial with the following docker-compose: https://github.com/Fiware/tutorials.Time-Series-Data/blob/master/docker-compose.yml
- B) HA deployment on Docker-Swarm (for production)
  - https://smartsdk-recipes.readthedocs.io/en/latest/data-management/quantumleap/readme/
- C) HA deployment on Kubernetes
  - (ask me how)

FIWARE

Step 1: Example (extract)

- QuantumLeap:
  - Needs to know where CrateDB is
- Crate:
  - Needs a volume to persist data
- Grafana:
  - Needs `crate-datasource` (or `PostgreSQL`) plugin

```
version: '3'
services:
  quantumleap:
    image: ${QL_IMAGE:-smartsdk/quantumleap}
    ports:
      - "8668:8668"
    depends_on:
      - mongo
      - orion
      - crate
    environment:
      - CRATE_HOST=${CRATE_HOST:-crate}
      - USE_GEOCODING=False

  crate:
    image: crate:${CRATE_VERSION:-3.0.5}
    command: crate -Clicense.enterprise=false -Cauth.host_based.enabled=false
      -Ccluster.name=democluster -Chttp.cors.enabled=true -Chttp.cors.allow-origin="*"
    ports:
      # Admin UI
      - "4200:4200"
      # Transport protocol
      - "4300:4300"
    volumes:
      - cratedata:/data

  grafana:
    image: grafana/grafana
    ports:
      - "3000:3000"
    environment:
      - GF_INSTALL_PLUGINS=crate-datasource,grafana-clock-panel,grafana-worldmap-panel
    depends_on:
      - crate
```

Step 2: “Connect” to Orion Context Broker

- Create a Subscription in Orion CB for the entities you are interested in!
  - A) Do it directly talking to Orion
    - https://fiware-orion.readthedocs.io/en/master/user/walkthrough_apiv2/index.html#subscriptions
  - B) Or ask QuantumLeap to do it for you
    - https://app.swaggerhub.com/apis/smartsdk/ngsi-tsdb/0.1.1#/input/reporter.reporter.subscribe
  - More info:
    - https://quantumleap.readthedocs.io/en/latest/user/#orion-subscription

Step 2: “Connect” to Orion

- POST orion/v2/subscriptions/
- Note **notification url** must be a valid url for Orion container.
- Note the inclusion of **dateModified**
- Use the same insertion headers
  - FIWARE-service
  - FIWARE-servicepath

```
{
  "description": "Notify QuantumLeap on luminosity changes on any Lamp",
  "subject": {
    "entities": [
      {
        "idPattern": "Lamp.*"
      }
    ],
    "condition": {
      "attrs": [
        "luminosity"
      ]
    }
  },
  "notification": {
    "http": {
      "url": "http://quantumleap:8668/v2/notify"
    },
    "attrs": [
      "luminosity"
    ],
    "metadata": ["dateCreated", "dateModified"]
  },
  "throttling": 1
}
```

Step 3: Get your data

- Check you are sending data to Orion
- Check Orion notifications did not fail
  - GET orion/v2/subscriptions
- Get your data
  - See example
- Nothing?
  - Checkout QuantumLeap logs!

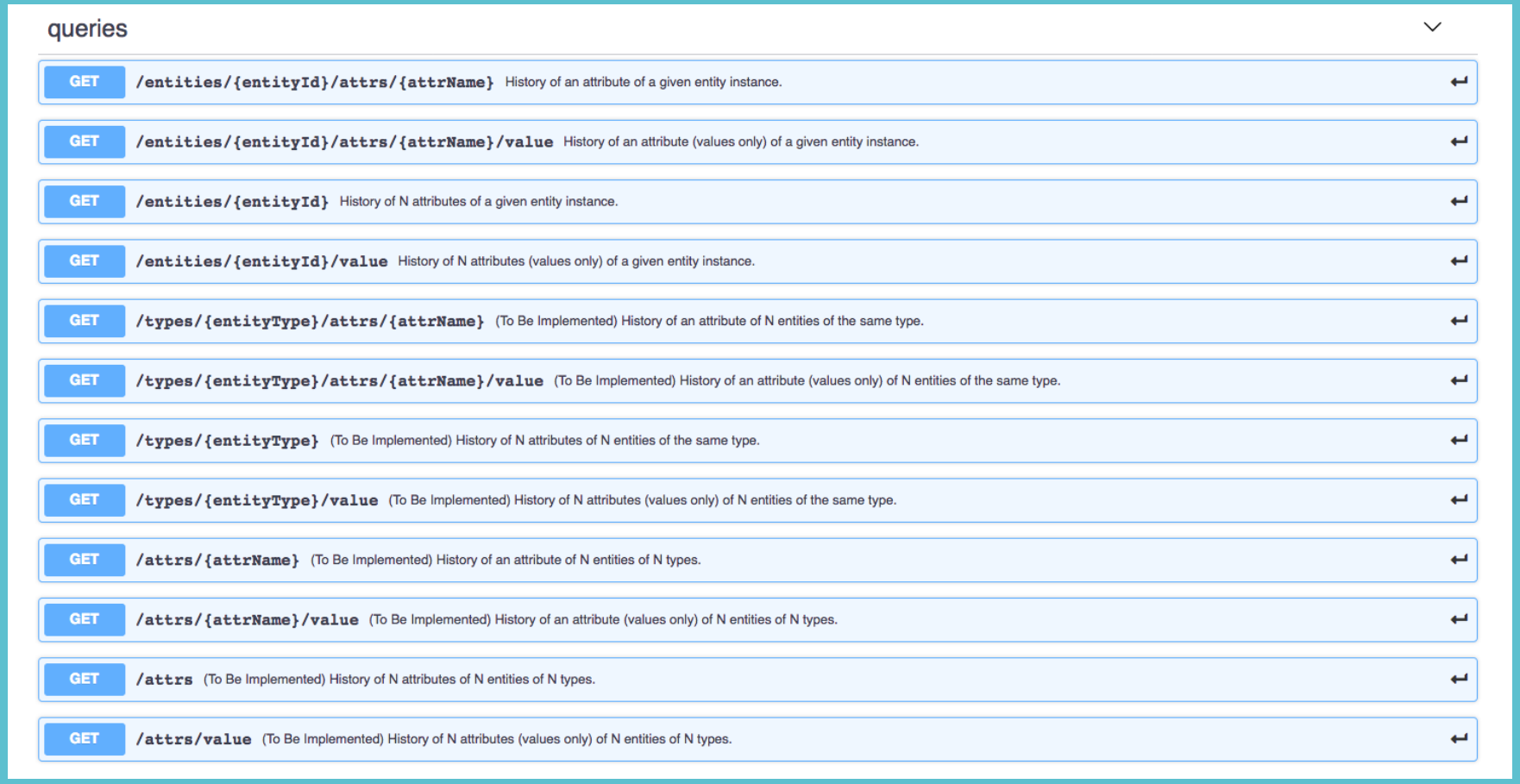
```
# REQUEST
curl -X GET \

'http://localhost:8668/v2/entities/Lamp:001/attrs/luminosity?=3&limit=3'
\
  -H 'Accept: application/json' \
  -H 'Fiware-Service: openiot' \
  -H 'Fiware-ServicePath: /'

# RESPONSE
{
    "data": {
        "attrName": "luminosity",
        "entityId": "Lamp:001",
        "index": [
            "2018-10-29T14:27:26",
            "2018-10-29T14:27:28",
            "2018-10-29T14:27:29"
        ],
        "values": [
            2000,
            1991,
            1998
        ]
    }
}
```

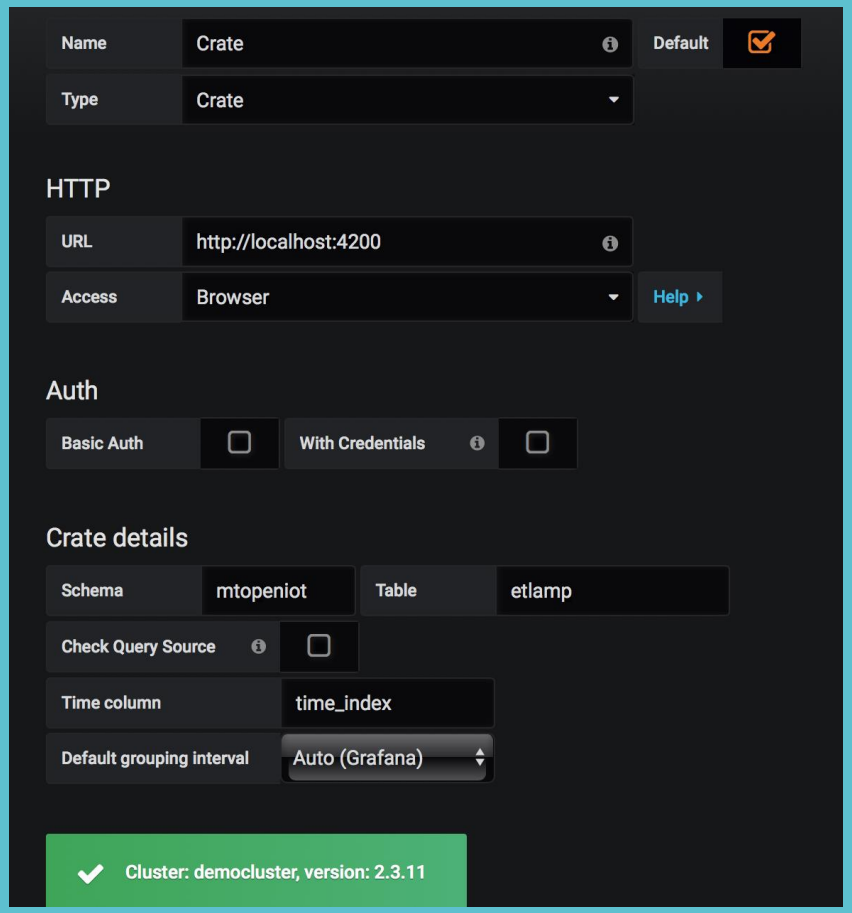
Step 3: Get your data

- Explore the API! (http://localhost:8668/v2/ui)

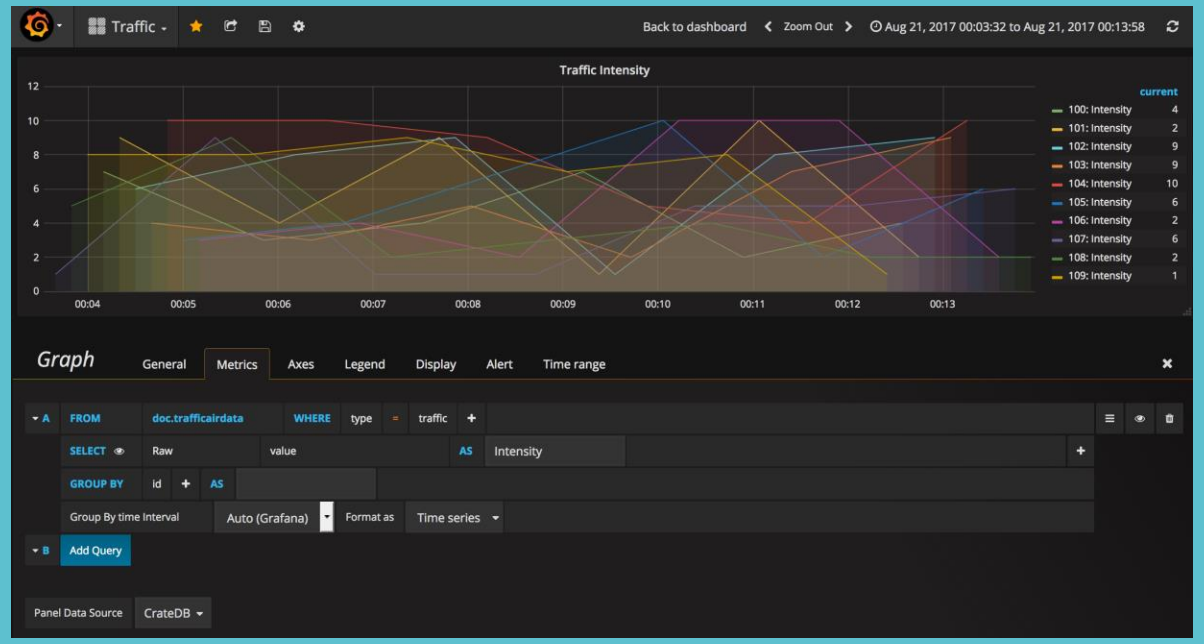


Step 4: Prepare your Dashboards

1. Create a Grafana data-source for each entity type



2. Create Grafana dashboards using your datasource



ROADMAP

https://github.com/smartsdk/ngsi-timeseries-api/milestones

FIWARE

QuantumLeap Roadmap

Version	Main Features	Exp. Release
0.5	Geo-queries support Improve handling of Open Street Maps	**24/12/18**
0.6	Virtual Entities support	**31/01/19**
0.6.1	Bugfixes and docs improvements	**15/02/19**
0.7	Visualization Helper (Auto-Dashboards and Access Control)	**29/03/19**
0.7.1	Bugfixes and docs improvements	**17/04/19**
1.0	Configuration API Complete Queries (/attrs & /types)	**17/05/19**
1.1	Bugfixes and docs improvements	**31/05/19**

https://github.com/smartsdk/ngsi-timeseries-api/milestones

Questions?

Useful QL Links

- TUTORIAL: https://fiware-tutorials.readthedocs.io/en/latest/time-series-data/index.html
- CODE: https://github.com/smartsdk/ngsi-timeseries-api
- DOCS: https://quantumleap.readthedocs.io/en/latest/
- API: https://app.swaggerhub.com/apis/smartsdk/ngsi-tsdb

Thank you!

http://fiware.org
Follow @FIWARE on Twitter