Making Sense of Sensors

Semantic Access to to Time Series Data

Manuela Corte Pause

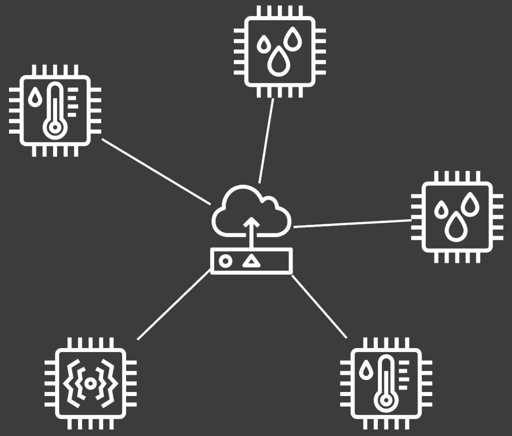
SFSCON, Bolzano, November 7th 2025

ONTOPIC

The IoT Data Challenge

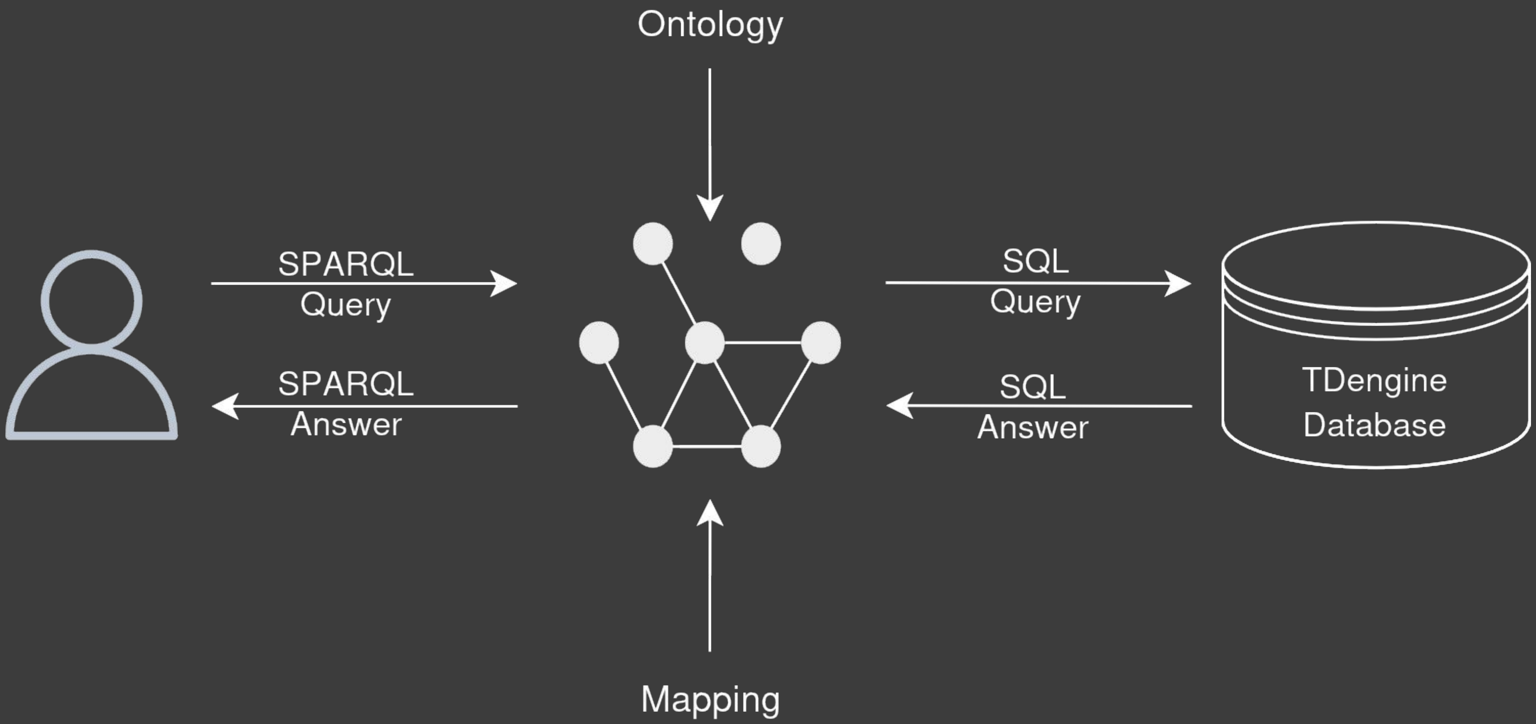
- IoT generates large amount of data

- Sensors from different vendors use different formats even if the data they refer to follow similar patterns



Solution: semantic layer over IoT data that provides a common representation

Our approach



ONTOPIC

TDengine

- Extremely fast open source time-series database

- **One table per device** approach (lock-free, append only writes)
- Logically unified structure for tables

```
indoor_sensors

ts                     TIMESTAMP
temperature            FLOAT
humidity               FLOAT
co2_ppm                INTEGER
voc_ppb                INTEGER
occupancy              INTEGER
----------------------------------
              TAGS
sensor_id, room_id, floor, building
----------------------------------
├── room_101_sensor
├── room_102_sensor
└── room_103_sensor
```

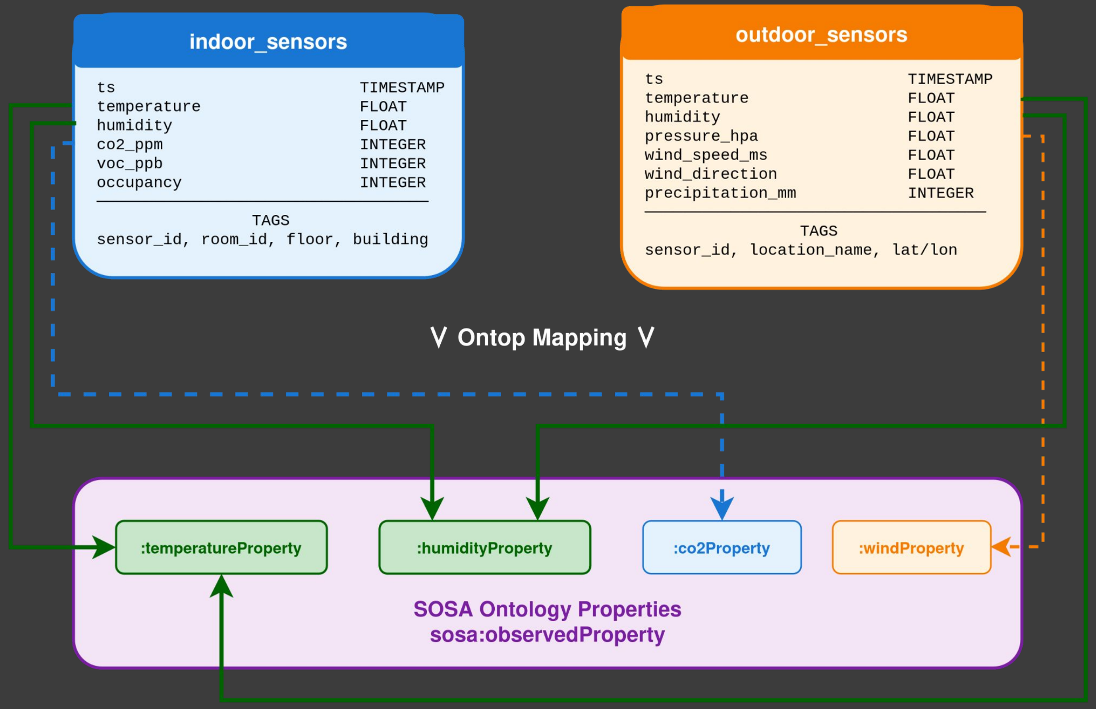
```
outdoor_sensors

ts                     TIMESTAMP
temperature            FLOAT
humidity               FLOAT
pressure_hpa           FLOAT
wind_speed_ms          FLOAT
wind_direction         FLOAT
precipitation_mm       INTEGER
----------------------------------
              TAGS
sensor_id, location_name, lat/lon
----------------------------------
├── roof_weather_station
└── parking_weather_station
```

ONTOPIC

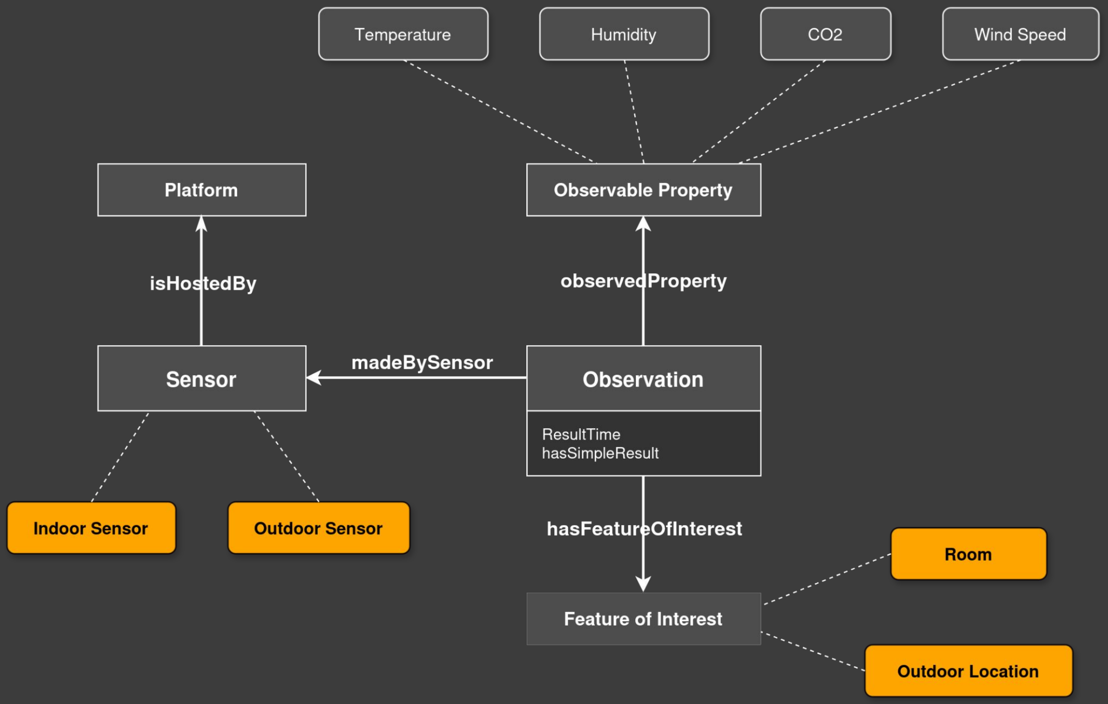
Ontop

- SPARQL to SQL translation
- Translation possible through **mapping**
- **Ontology** as shared vocabulary



ONTOPIC

SOSA Ontology



- Widely used ontology
- **Interoperability** across IoT systems
- Captures not just data values, but measurement context

ONTOPIC

Putting the Pieces Together

```
PREFIX sosa: <https://www.w3.org/TR/vocab-ssn/>
PREFIX : <http://example.org/building/>

SELECT ?location ?humidity ?time
WHERE {
    ?obs sosa:observedProperty :humidityProperty ;
         sosa:hasFeatureOfInterest ?location ;
         sosa:hasSimpleResult ?humidity ;
         sosa:resultTime ?time .

    FILTER(?humidity > 46.0)
}
ORDER BY DESC(?humidity)
```

- The semantic layer can be queried in SPARQL
- Agnostic to the structure of the underlying data source

	location	humidity	time
1	http://example.org/building/Rooftop	"65.5"^^xsd:float	"2025-10-27T14:00:00.000"^^xsd:dateTime
2	http://example.org/building/Rooftop	"64.8"^^xsd:float	"2025-10-27T14:05:00.000"^^xsd:dateTime
3	http://example.org/building/room_101	"47.8"^^xsd:float	"2025-10-27T14:05:00.000"^^xsd:dateTime

ONTOPIC

More Complex Query

```
PREFIX sosa: <https://www.w3.org/TR/vocab-ssn/>
PREFIX : <http://example.org/building/>

SELECT ?time (AVG(?indoorTemp) AS ?avgIndoor) (AVG(?outdoorTemp) AS ?avgOutdoor)
       ((AVG(?indoorTemp) - AVG(?outdoorTemp)) AS ?difference)
WHERE {
  ?indoorObs a :IndoorObservation ;
             sosa:observedProperty :temperatureProperty ;
             sosa:hasSimpleResult ?indoorTemp ;
             sosa:resultTime ?time .

  ?outdoorObs a :OutdoorObservation ;
              sosa:observedProperty :temperatureProperty ;
              sosa:hasSimpleResult ?outdoorTemp ;
              sosa:resultTime ?time .
} GROUP BY ?time
ORDER BY ?time
```

Similarly to SQL, SPARQL provides more complex functionalities

	time	avgIndoor	avgOutdoor	difference
1	"2025-10-27T14:05:00.000"^^xsd:dateTime	"23.100000381469727"^^xsd:float	"15.5"^^xsd:float	"7.600000381469727"^^xsd:float
2	"2025-10-27T14:00:00.000"^^xsd:dateTime	"22.5"^^xsd:float	"15.199999809265137"^^xsd:float	"7.300000190734863"^^xsd:float

Conclusion

Benefits

- Great performance with added semantics
- Data stays in the original database
- All technologies are open source and easily available

Challenges

- Additional layer of abstraction
- Ontop - TDengine integration is still a work in progress

Thanks for your attention!

Questions?

ONTOPIC