InterSensor Service

Managing Heterogeneous Sensor Observations in Smart Cities

Kanishk Chaturvedi, Thomas H. Kolbe

Chair of Geoinformatics
Technische Universität München

Geospatial Sensor Webs Conference
Münster, Germany
September 4, 2018

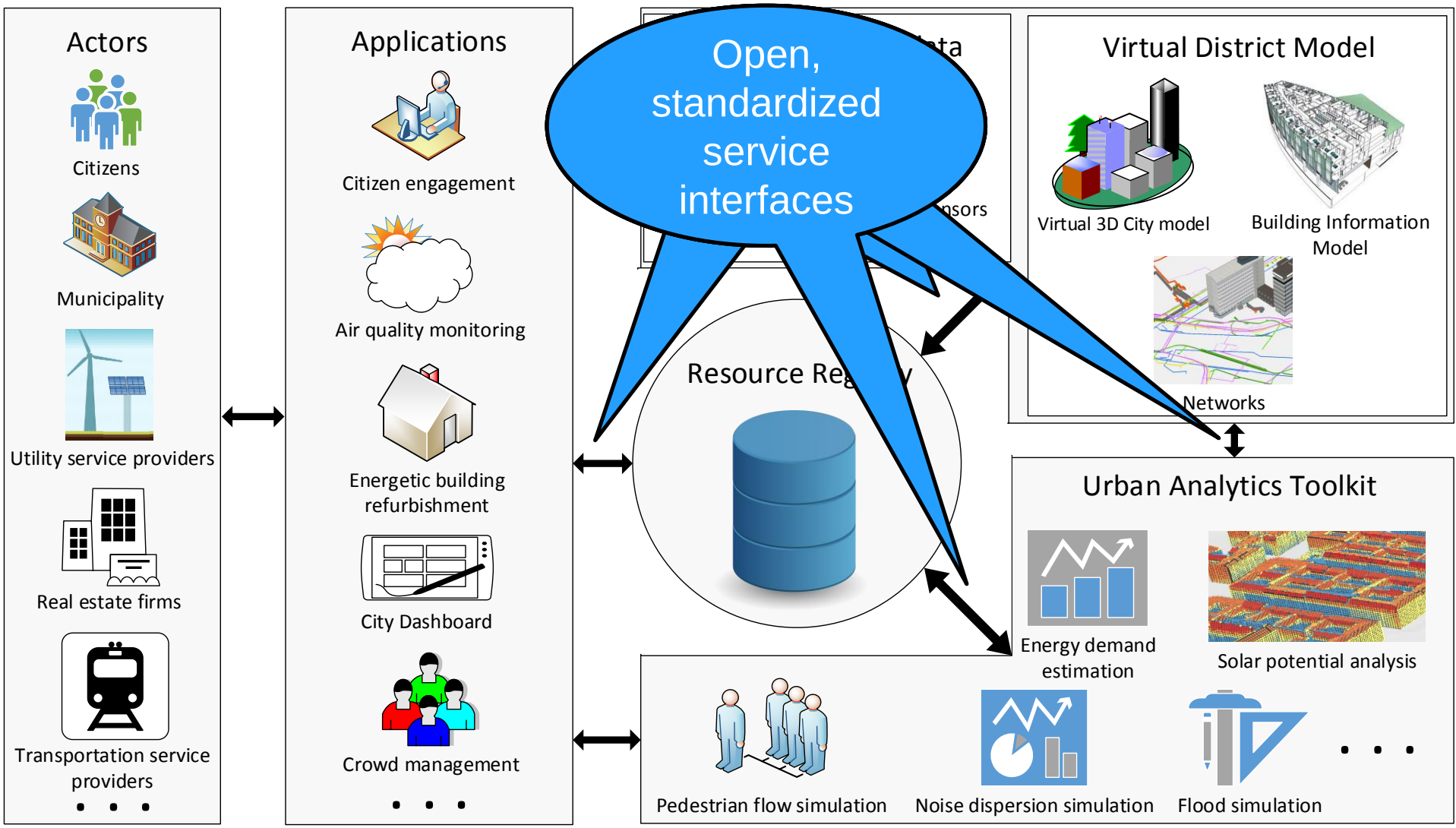
Smart Cities and Interoperability

- Most smart city projects involve distributed systems
  - With different stakeholders (e.g. owners, operators, solution providers, citizens, visitors), agents and communities
  - Having different interest, goals, and tasks
  - And different roles and rights
- In distributed environment, it is unlikely that stakeholders
  - would be willing to use a common platform
  - Would be willing to share proprietary data to others
- Key requirement
  - Standardized services/interfaces
  - Standardized information models and data formats
- **Interoperability plays a key role!!**

Smart City Initiatives focusing on Interoperability

- Future City Pilot Phase 1
  www.opengeospatial.org/projects/initiatives/fcp1
- ESPRESSO
  www.opengeospatial.org/projects/initiatives/espresso
- Smart City Interoperability Reference Architecture (SCIRA)
  www.opengeospatial.org/projects/initiatives/scira
- City Enabler for Digital Urban Services (CEDUS)
  www.cedus.eu/
- Smart District Data Infrastructure (SDDI)
  www.gis.bgu.tum.de/en/projects/smart-district-data-infrastructure/

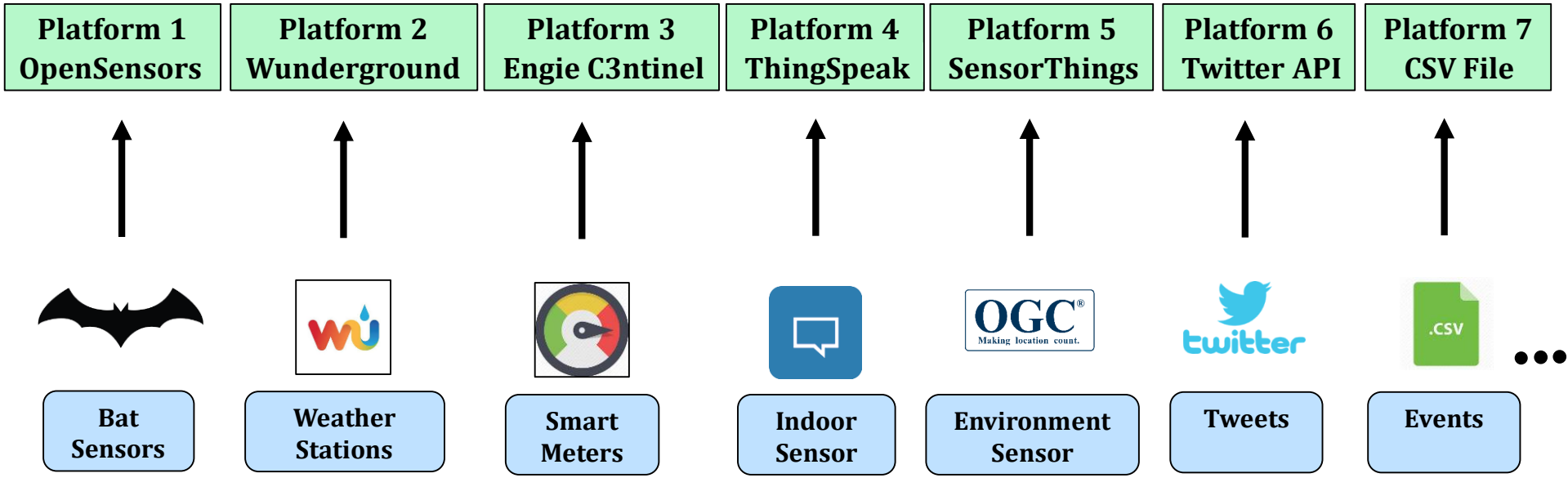
Smart District Data Infrastructure (SDDI)



SDDI - Queen Elizabeth Olympic Park

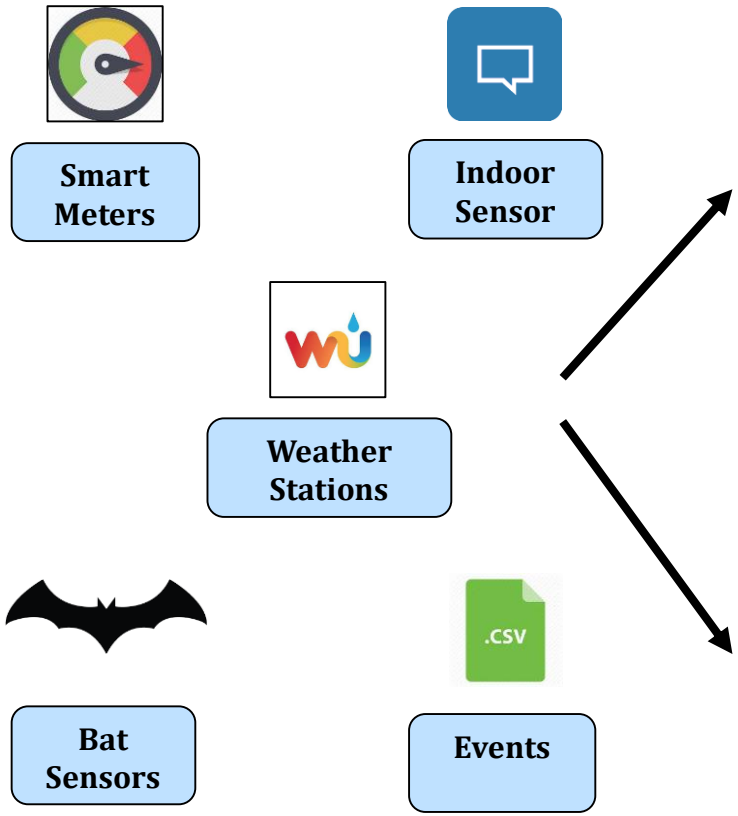
- A sporting complex built for the 2012 Summer Olympics
- Four key themes:
  - Resource Efficient Buildings
  - Energy Systems
  - Smart Park / Future Living
  - Data Architecture and Management
- Partners
  - Technical University of Munich
  - Imperial College London
  - virtualcitySYSTEMS GmbH
  - London Legacy Development Corporation

Sensors/IoT are an integral part of SDDI - QEOP



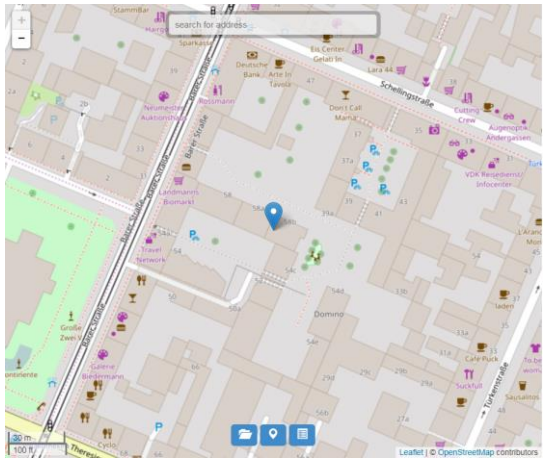
- Different data sources
  - Are used for different purposes/applications
  - have platforms and APIs
  - have data formats and interfaces
- How to work with all of data sources in an interoperable way?

Solution - OGC Sensor Web Enablement



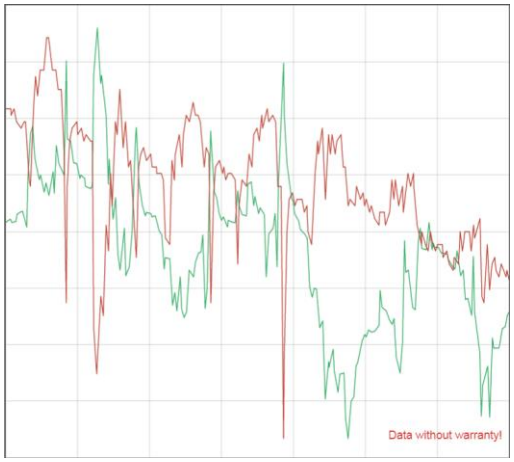
Sensor Description

```
OGC SensorML
<sensorDescription>
   <!--Identifier-->
   <!--Geographic position-->
   <!--List of properties-->
   <!--Sensor Owner-->
   <!--Other metadata -->
</sensorDescription>
```



Observations

```
OGC Observations & Measurements
<OM_Observation>
  <!--Observation property-->
  <!--Observation time-->
  <!--Observation value-->
</OM_Observation>
```



OGC Sensor Web Enablement Interfaces

▶ OGC Sensor Observation Services (SOS)

- Open standard, part of OGC Sensor Web Enablement (SWE)
- Allows querying real-time sensor data and sensor data timeseries.
- Observation responses are encoded according to the O&M standard

▶ OGC SensorThings API

- Very lightweight standard to interconnect the IoT devices, data and applications over the web
- Built on the OGC SWE and O&M standards
- Provides REST services and compact data encodings in JSON format

▶ 52° North Timeseries API

- REST-ful web binding to the OGC Sensor Observation Service.
- Allows querying and visualizing sensor locations and their observations on the so-called Helgoland client

OGC SWE is the way forward!

Application 1
OGC SOS

Application 2
OGC SensorThings API

Sensor Data Visualization
52° North Timeseries API

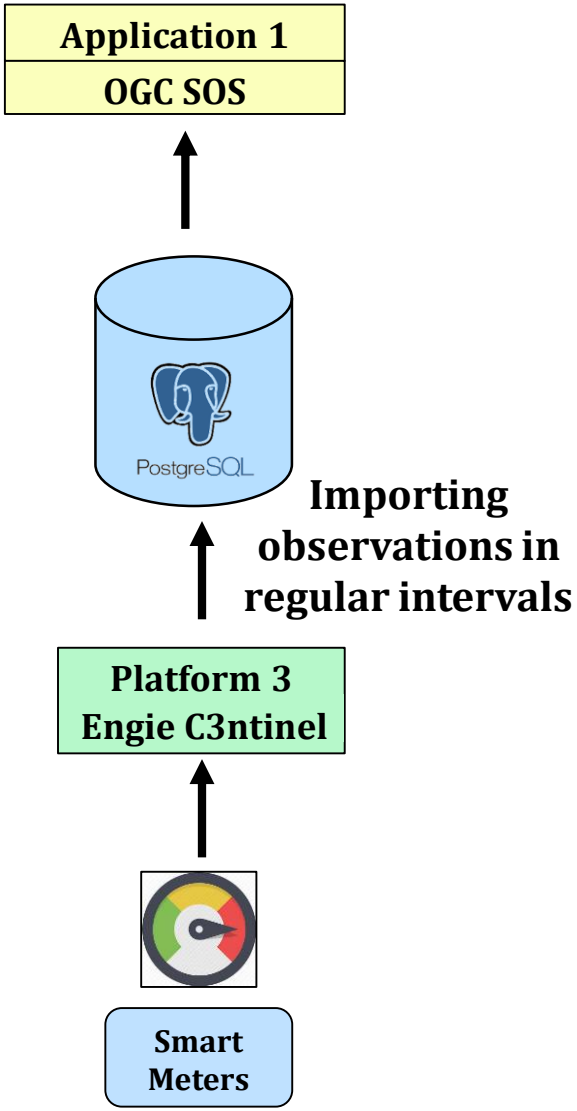
How to encode the data according to the standardized interfaces?

Platform 1 OpenSensors	Platform 2 Wunderground	Platform 3 Engie C3ntinel	Platform 4 ThingSpeak	Platform 5 SensorThings	Platform 6 Twitter API	Platform 7 CSV File
Bat Sensors	Weather Stations	Smart Meters	Indoor Sensor	Environment Sensor	Tweets	Events

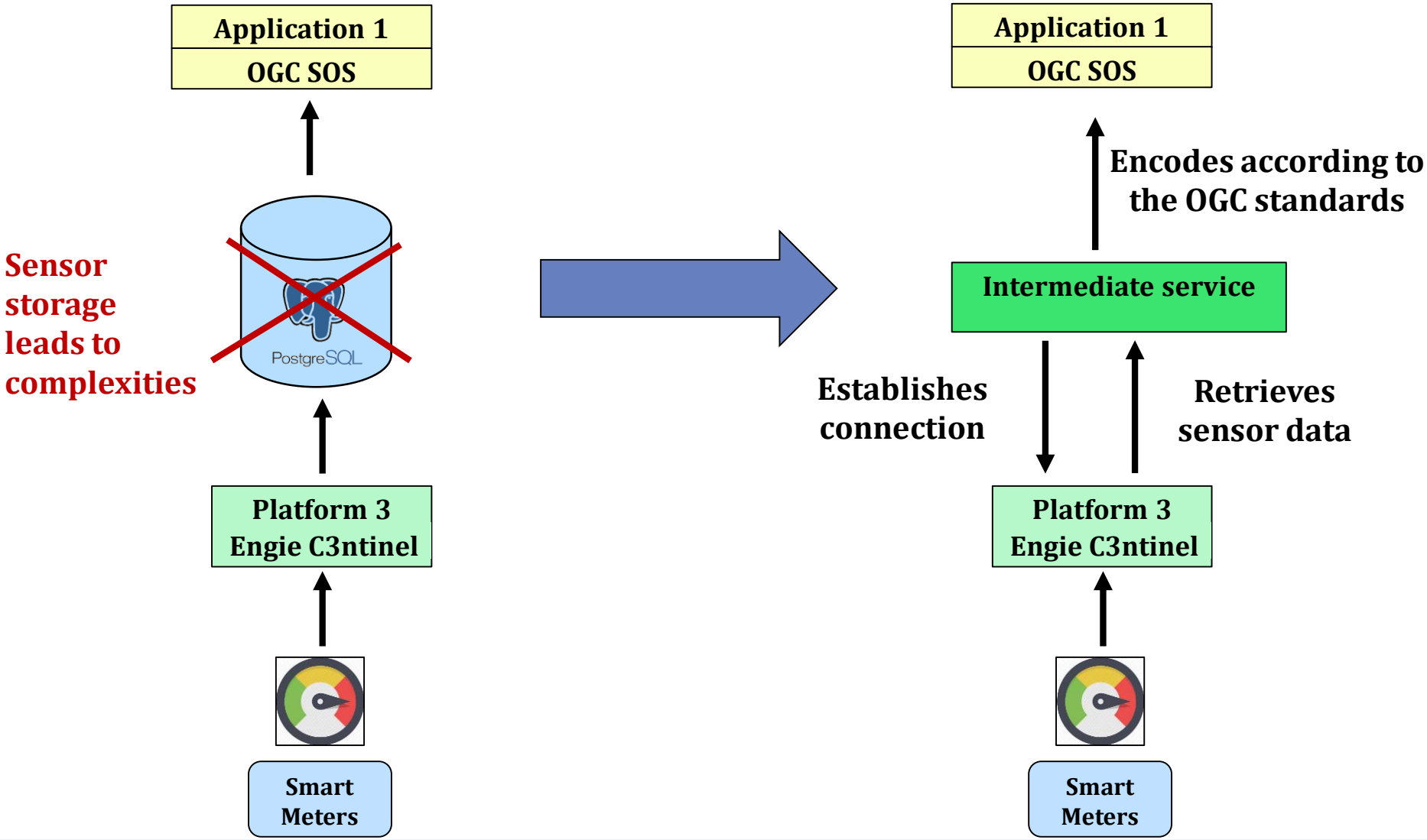
...

Setting up OGC SWE Interfaces

- The implementations always require a data storage for retrieval of sensor observations
- The timeseries data always needs to be imported to the database (e.g. SOS database)
- Challenges
  - Not all of the stakeholders would be willing to inject their data into another data storage for sensor web
  - Requires regular maintenance for the sensor web data storage
  - Involved complexity in moving the infrastructure from one server to another, or to cloud



OGC SWE Interfaces in Distributed Scenarios

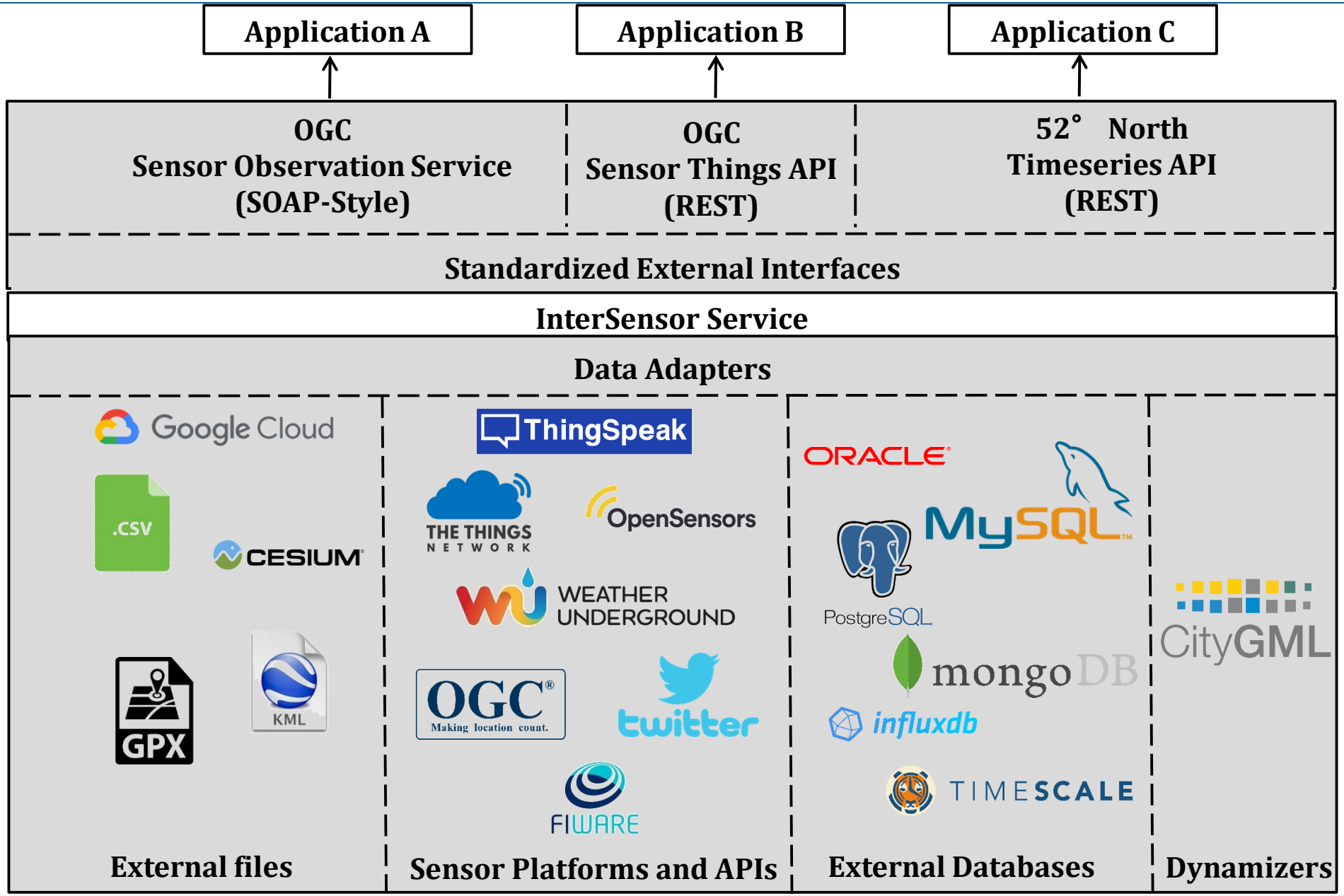


InterSensor Service

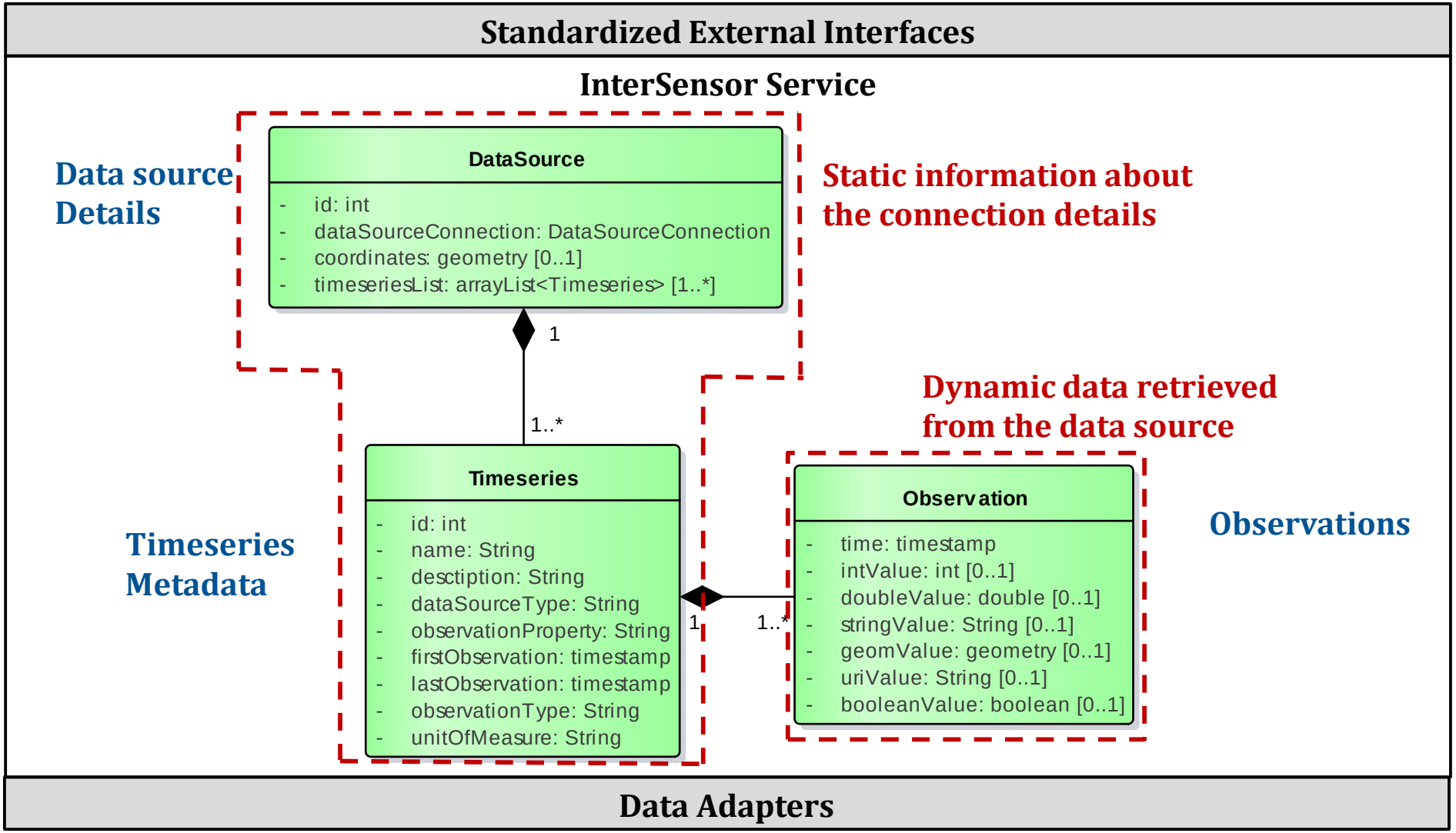
- Very basic and lightweight service
- InterSensor Service operates on arbitrary data sources
  - External files (CSV, Excel sheets)
  - Cloud based services (Google Spreadsheet, Google Fusion Tables)
  - External databases (JDBC connections)
  - External IoT Platforms (Thingspeak, OpenSensors etc..)
- Encodes data “on-the-fly” according to standardized interfaces
  - OGC Sensor Observation Service (Core profile)
  - OGC SensorThings API
  - 52° North Timeseries API
- Based on the Spring Framework
- Free and Open Source

  www.github.com/tum-gis/InterSensorService

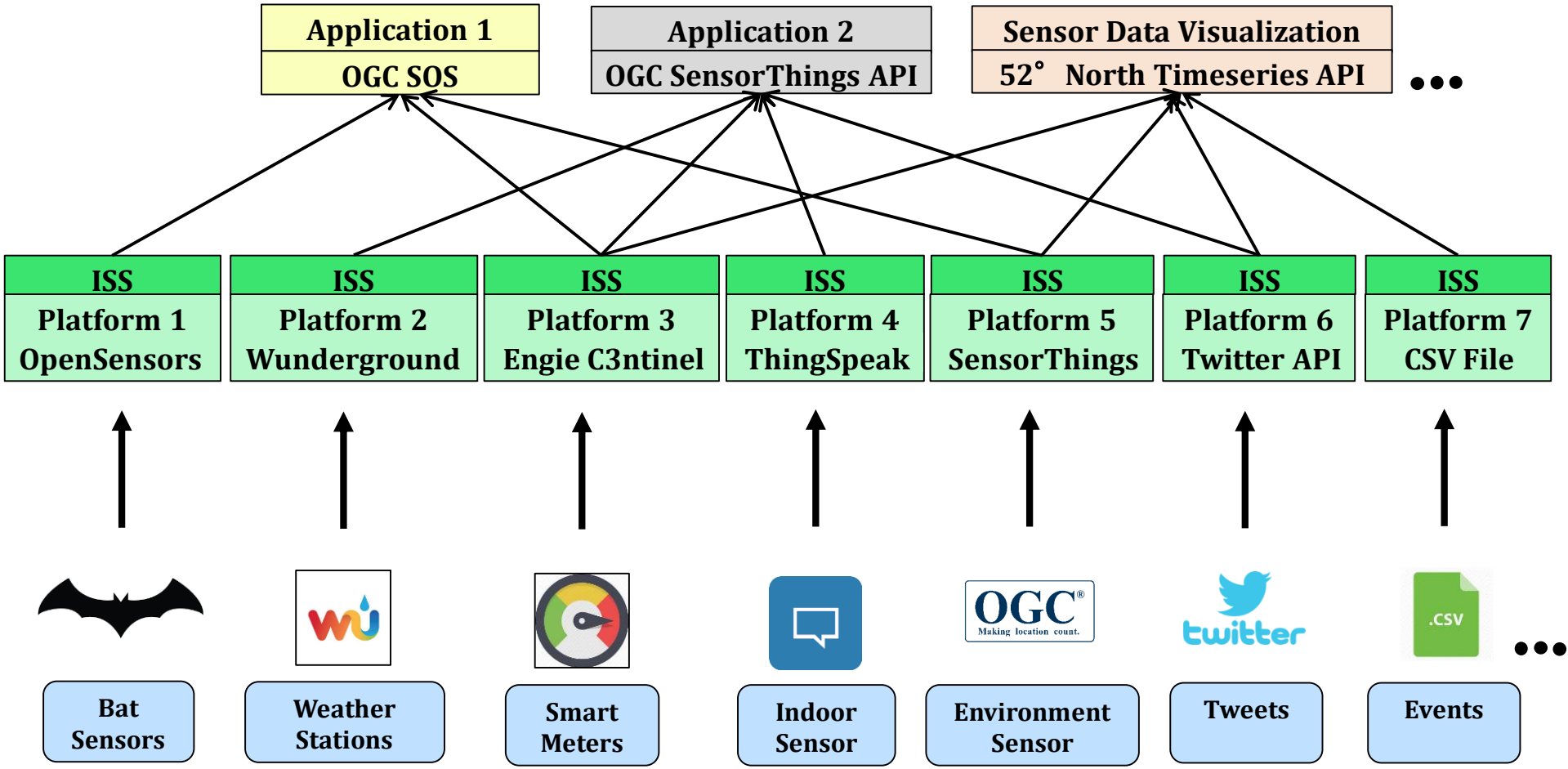
  www.intersensorservice.org (Coming soon!!)


Application A
Application B
Application C
OGC
Sensor Observation Service
(SOAP-Style)
OGC
Sensor Things API
(REST)
52° North
Timeseries API
(REST)
Standardized External Interfaces
InterSensor Service
Data Adapters
Google Cloud
.CSV
CESIUM
GPX
KML
ThingSpeak
THE THINGS NETWORK
OpenSensors
WEATHER UNDERGROUND
OGC
Making location count.
twitter
FIWARE
ORACLE
MySQL
PostgreSQL
mongoDB
influxdb
TIMESCALE
CityGML
External files
Sensor Platforms and APIs
External Databases
Dynamizers

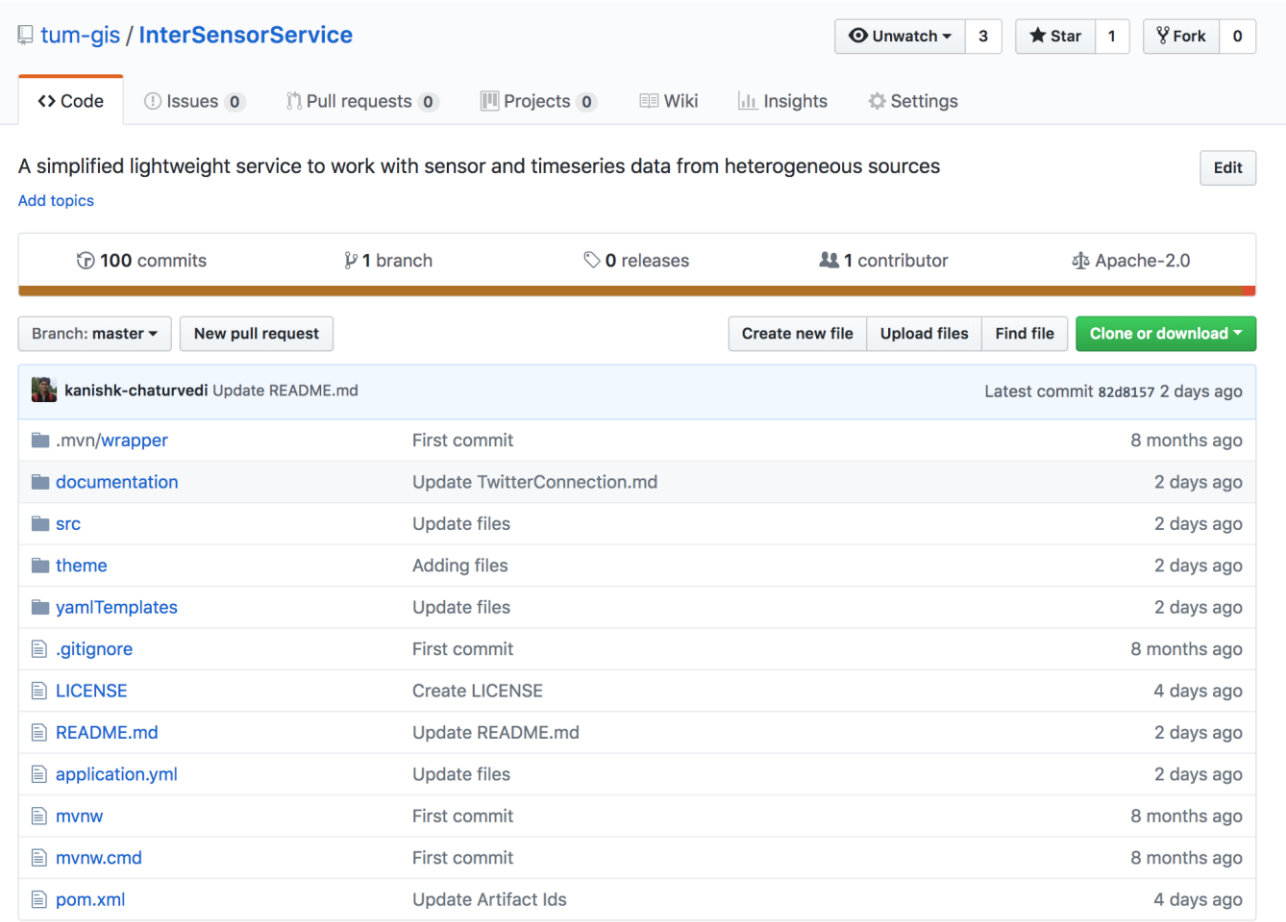
Data Model


Standardized External Interfaces
InterSensor Service
Data source Details
DataSource
- id: int
- dataSourceConnection: DataSourceConnection
- coordinates: geometry [0..1]
- timeseriesList: arrayList<Timeseries> [1..*]
Static information about the connection details
1
1..*
Dynamic data retrieved from the data source
Timeseries Metadata
Timeseries
- id: int
- name: String
- desctiption: String
- dataSourceType: String
- observationProperty: String
- firstObservation: timestamp
- lastObservation: timestamp
- observationType: String
- unitOfMeasure: String
1
1..*
Observation
- time: timestamp
- intValue: int [0..1]
- doubleValue: double [0..1]
- stringValue: String [0..1]
- geomValue: geometry [0..1]
- uriValue: String [0..1]
- booleanValue: boolean [0..1]
Observations
Data Adapters

Applying the InterSensor Service in QEOP

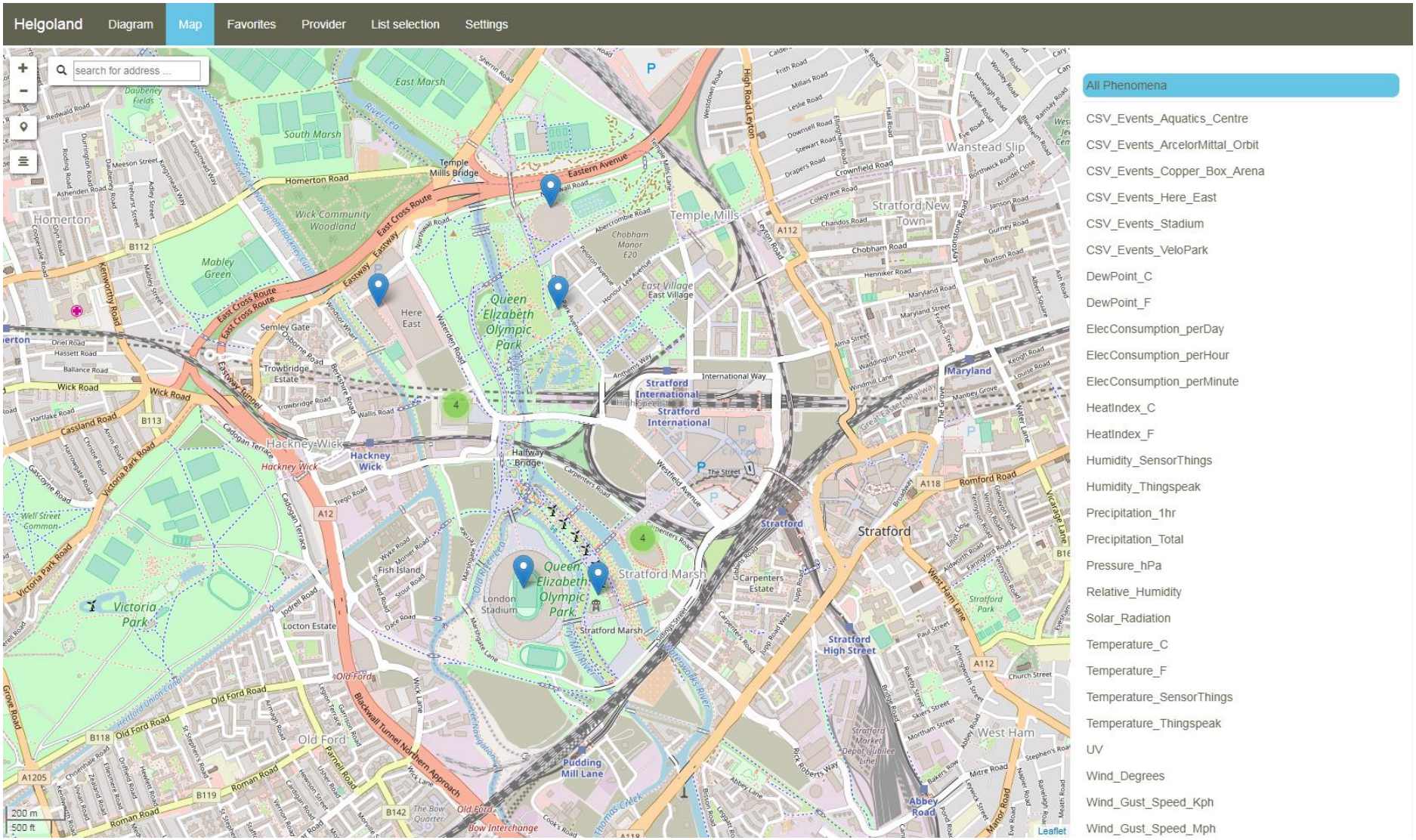


Getting started with the InterSensor Service



www.github.com/tum-gis/InterSensorService

Demo - Multiple Sensor locations on Helgoland client



Demo - Multiple Sensor observations on Helgoland client

Demo – Visualizing Geo-tagged tweets with CityGML objects

Future Work

- **Supporting more data sources**
  - Timeseries and Non-relational Databases
  - Cloud based systems (Google Fusion Tables)
  - Moving objects (such as GPS Exchange Formats, KML, CZML)
- **Complete support for standardized interfaces**
  - Development of the Service Provider Interface (SPI) for the 52° North Timeseries API
  - Complete querying capabilities of the SensorThings API
- **Support of Docker containers**
  - To quickly set up instances of the service and move them to the Cloud environment
- **Supporting write access?**
- **InterSensor Service as an additional service**
  - To access dynamic attributes with the CityGML Dynamizers and 3DCityDB