Safe Cities

Building Urban Safety

U.PORTO BOSCH

Workshop

FIWARE

25 February

9h – 13h

COMPETE 2020 PORTUGAL 2020

UNIÃO EUROPEIA

Fundo Europeu de Desenvolvimento Regional

Agenda

1. FIWARE Platform
    - Contexts
    - Components
2. Interactions & APIs
3. FIWARE Architecture
    - Generic Architecture
    - Supported Models (Sigle-Tenant, Multi-Tenant, Federated)
    - Deployments (Local, Cloud, Hybrid)

4. FIWARE Applications
    - Design
    - Data Flow
5. Smart-City Applications (Demonstration)

THE FIWARE PLATFORM

Safe Cities
Building Urban Safety

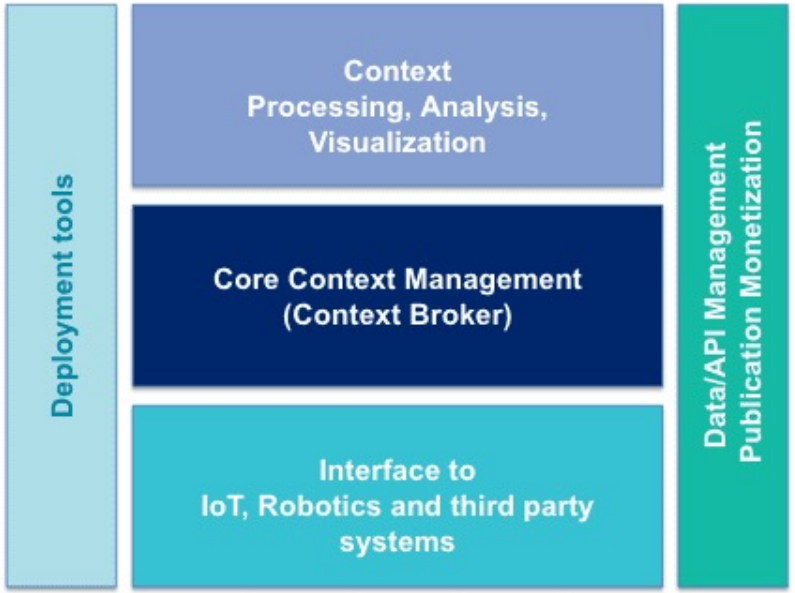
U.PORTO
BOSCH

SP5
FIWARE

- Open-source initiative aimed at defining a set of standards for ***context data management***
  - For producing, gathering, publishing and consuming **context data** and information at scale

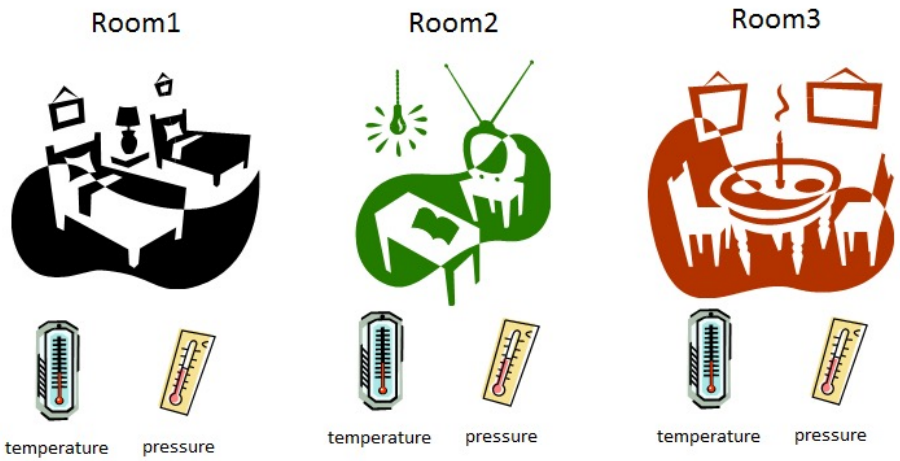
- Defines a set of open standards, referred to as **Generic Enablers**, APIs and reference implementations
  - Connecting to Internet of Things Devices (e.g., sensors and actuators)
  - Interacting with devices (e.g., acquiring and distributing data)
  - Processing and analysing data, Big Data, and/or real-time media
  - Publishing data and resulting information (i.e., processed data)
  - Monetizing from data and information

FIWARE
OPEN APIs FOR OPEN MINDS



SP5
FIWARE - Context Data

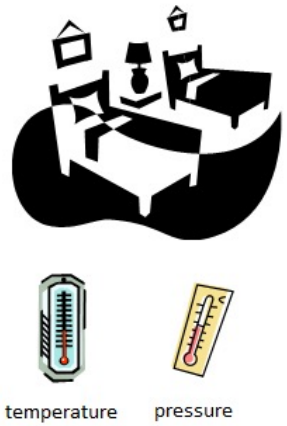
- Abstraction that binds data with its associated context
  - The context in which this data is acquired or associated to
- Allows to model and access to data independently from the source or application that will “eventually” consume it
  - Without forcing a rigid data model (e.g., a database schema)
- Context Data can be characterized with additional meta-data
  - Represented by attributes that characterize the context of the data collecting entities (i.e., sensors)
  - Applications can access data by different querying schemas



SP5

FIWARE - Context Data - Example

Room1




```
{
  "id": "Room1",
  "type": "Room",
  "temperature":
  {
     "value": 23.7,
     "type": "Float"
  },
  "pressure":
  {
     "value": 720,
     "type": ”Number"
  }
}
```

Room2


```
{
  "id": "Room2",
  "type": "Room",
  "temperature":
  {
     "value": 21.2,
     "type": "Float"
  },
  "pressure":
  {
     "value": 711,
     "type": "Number"
  }
}
```

Room4


```
{
  "id": "Room4",
  "type": "Room",
  "pressure":
  {
     "value": 718,
     "type": "Number"
  }
}
```

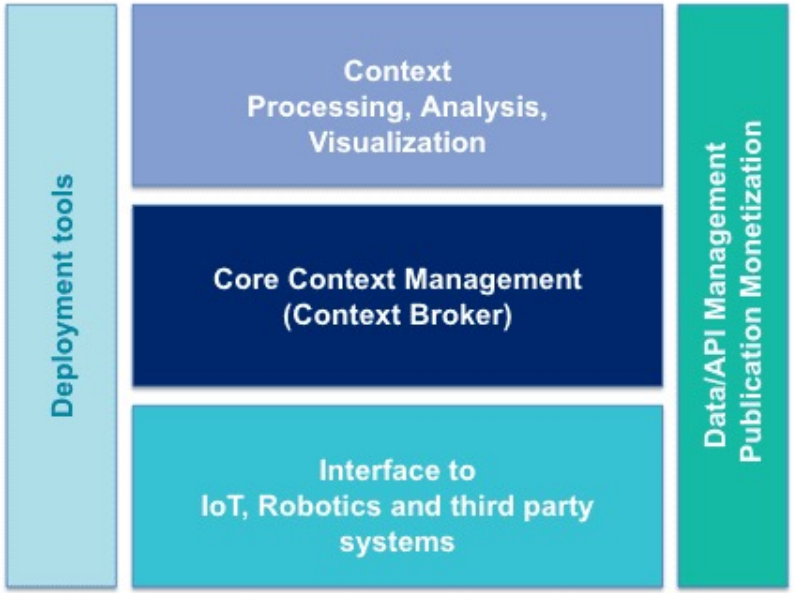
SP5
FIWARE - Generic Enablers

- Set of **standards** and **components** that provide functionalities for devices, systems and applications to interact with FIWARE
  - Allow the ecosystem to provide some functionality

- Generic Enablers (GEs) include components for
  - Virtualizing physical devices/systems
  - Produce and/or consume data/information
  - Publish and routing data/information to interested parties
  - Provide security functionalities

- Follows a layered or stack approach, where each layer/component provides a REST API for the other components

FIWARE
OPEN APIs FOR OPEN MINDS

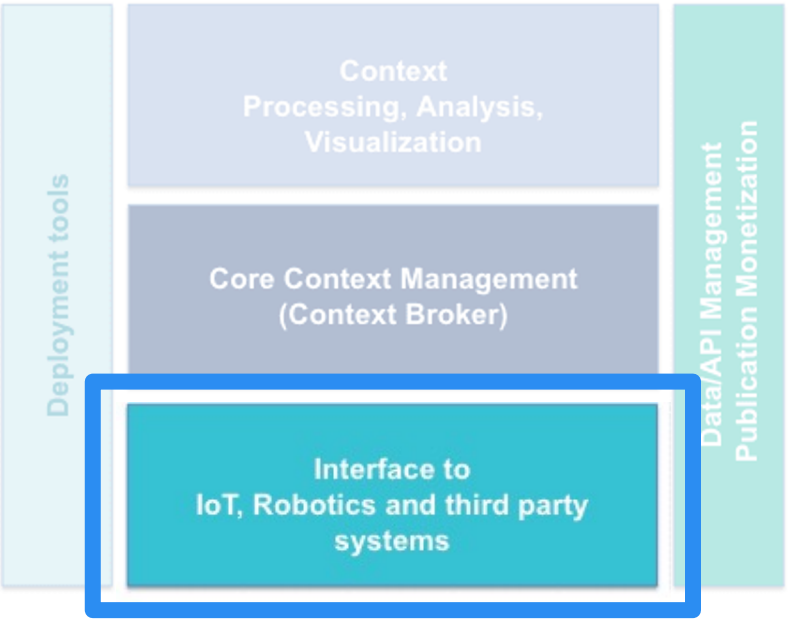


SP5
FIWARE – IoT Agent Generic Enablers

- **IoT environments** are synonymous with **heterogeneity** due to the lack of globally accepted standards
  - **Heterogeneity** in terms of devices, protocols, communication technologies, etc.

- IoT Agent GEs provide an interface with Internet of Things devices, Robots and Third-party systems
  - Translating device-specific protocols into the FIWARE standard data exchange model, i.e., context information
  - Abstracting from intrinsic heterogeneity, offering virtualized views of the respective device/system, accessible through a REST API
- Allows gathering context information and/or trigger actuations in response to context updates

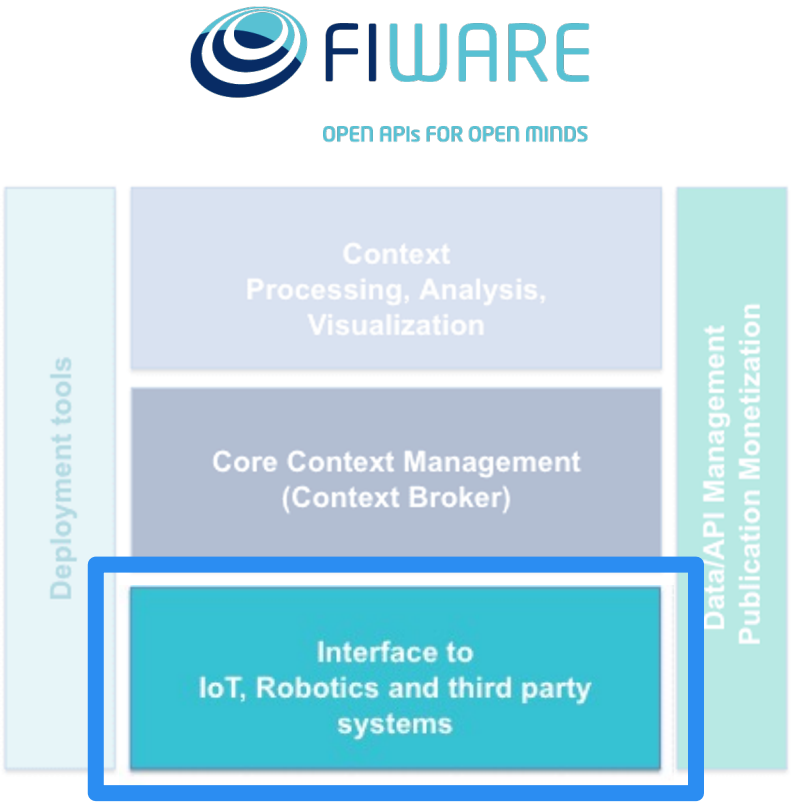
FIWARE
OPEN APIs FOR OPEN MINDS



SP5
FIWARE – IoT Agent Generic Enablers

- FIWARE **offers** IoT Agent GEs implementations supporting
  - JSON (Javascript Object Notation)
  - Ultralight 2.0 protocol with AMQP, HTTP and MQTT transports
  - Low Range Wide-area Network (LoRaWaN)
  - Light-Weight Machine2Machine protocol (COAP)

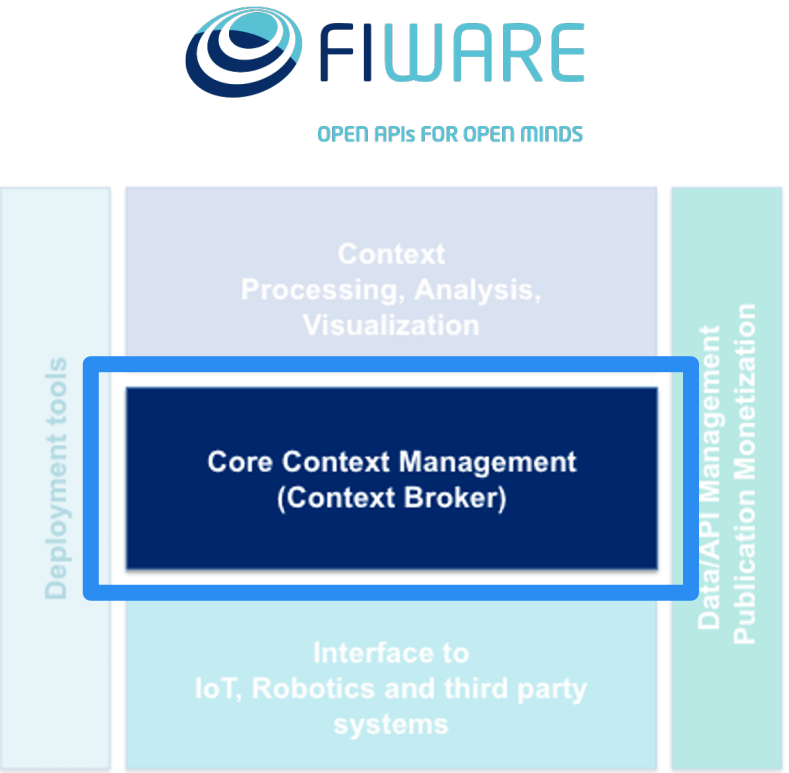
- Offer a framework library for developing Custom IoT Agent



SP5

FIWARE – Context Management Generic Enabler

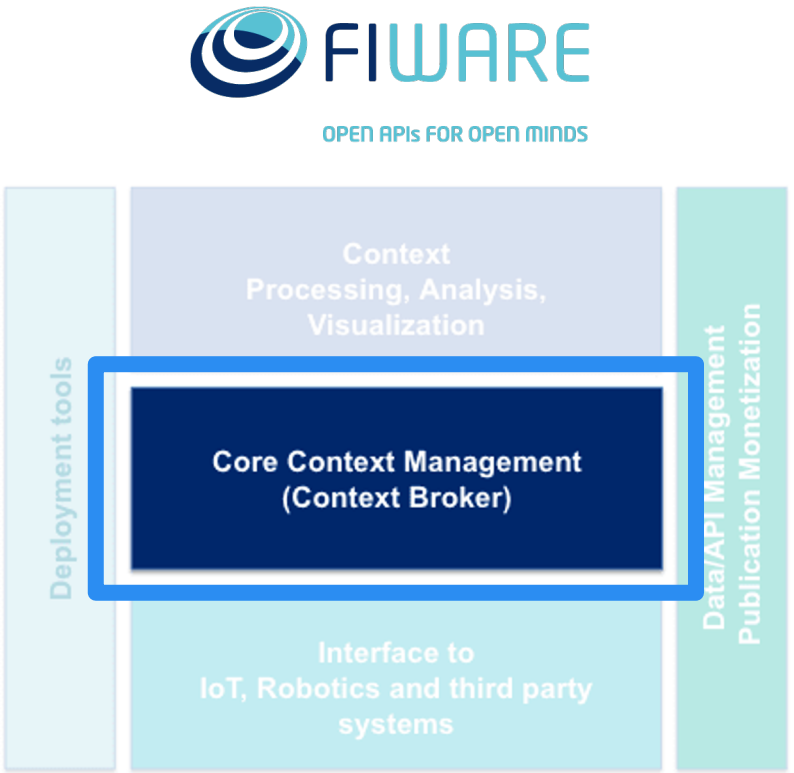
- Context Broker GE is the **core** and **mandatory component** of any "Powered by FIWARE" platform or application
  - It provides management of context information in a distributed and large-scale manner
- Context Broker GE provides support for registering, querying, updating or subscribing to changes on context information
  - Offers a Publish/Subscribe interaction model for exchanging context information between interested parties
  - Accessible through a REST API (referred to as FIWARE NGSI v2 API)
- FIWARE also provides GEs for persistently storing context information and connecting to Big-Data processing infrastructures



SP5

FIWARE – Context Management Generic Enabler

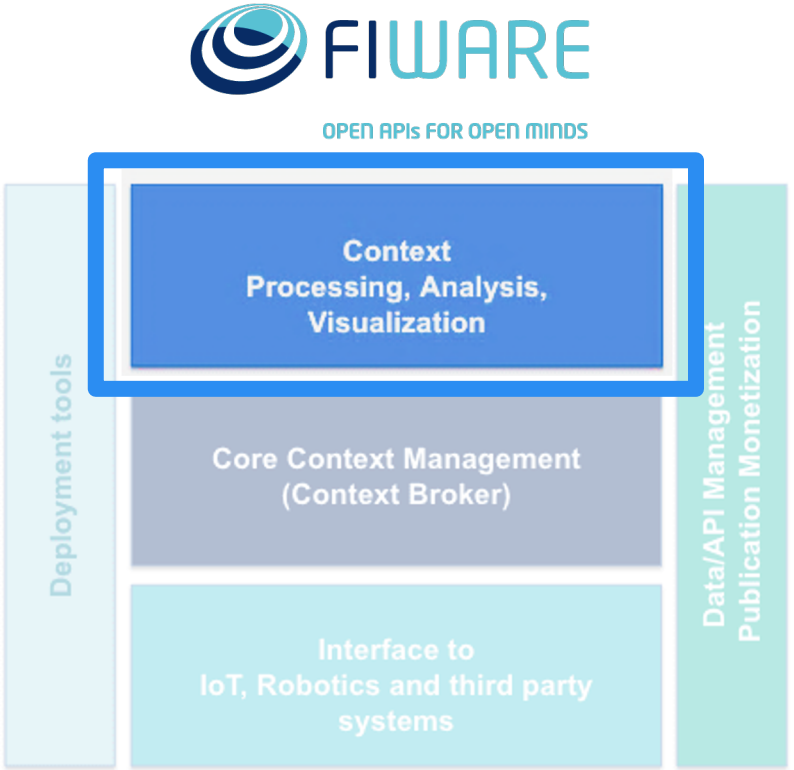
- Comet, Cygnus, Draco and QuantumLeap GEs offer short- and long-term history storage support for context information
  - Comet offers STH storage support (typically months), supported by MongoDB database management system (DBMS)
  - Cygnus offers LTH storage support to DBMS, including PostgreSQL, MySQL, MongoDB and AWS DynamoDB
  - Draco offers storage designed for flow-based programming, based on Apache NiFi
  - QuantumLeap offers supports for connecting to time-series based DBMS, including CrateDB and Timescale
- Cygnus and Cosmos GEs offer sinks for Big-Data platforms
  - Including Hadoop, Storm, Spark and Flink



SP5

FIWARE – Context Processing and Analysis Generic Enabler

- FIWARE offers GEs for processing, analysing and visualizing context information
- Wirecloud GE offers a web mashup platform to develop operational dashboards
- FogFlow GE is a distributed execution framework to support dynamic processing flows over cloud and edges
- Perseo GE introduces Complex Event Processing (CEP) defined using a rules-based system



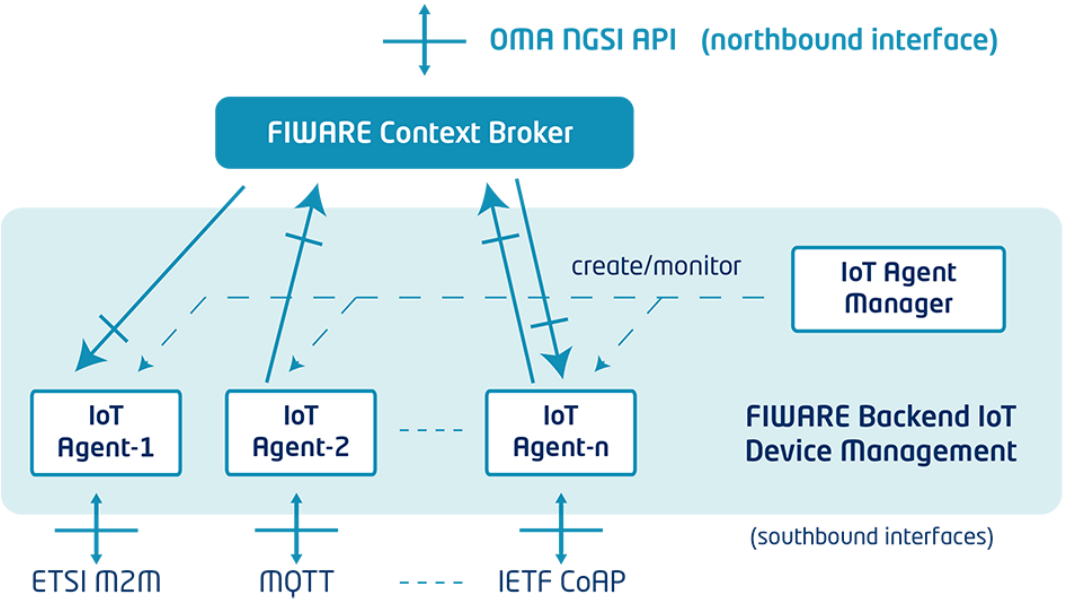
INTERACTIONS & APIs

Safe Cities
Building Urban Safety

U.PORTO
BOSCH

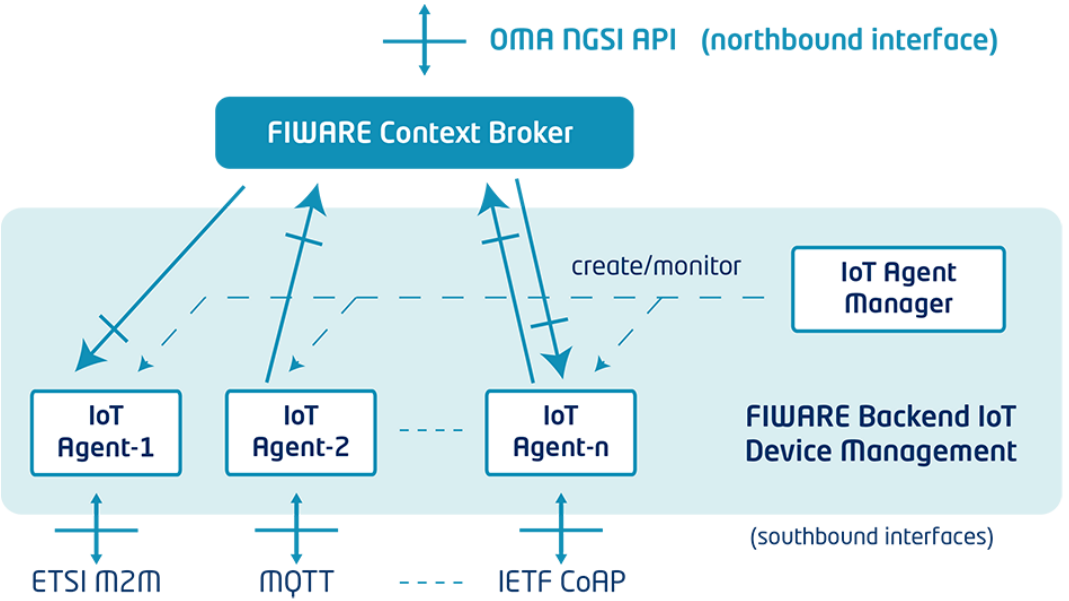
SP5
FIWARE – IoT Agent

- **IoT Agent** provides a common framework for connecting IoT devices
  - Offers standardized mapping for devices and data, provisioning, etc.
- IoT Agents abstract device specific protocols and virtualize devices
- API
  - Device Provisioning
  - Service Provisioning



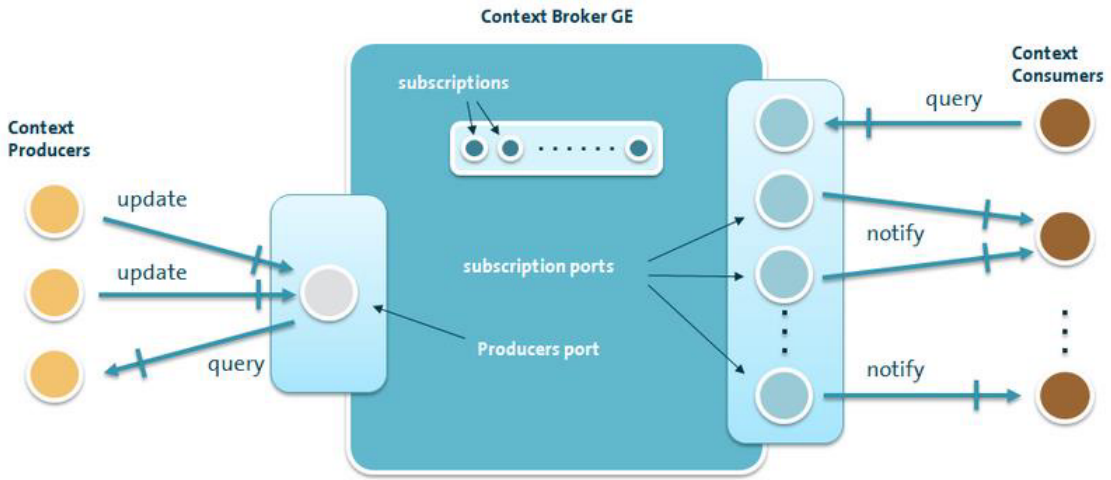
SP5
FIWARE – IoT Agent – Provisioning

- Device Provisioning provides support for mapping a single device with the Context Broker
  - Registering all the information about the device
  - Required attributes: service name, service path, device id, name and type
- Service Group Provisioning provides support for mapping multiple devices to specific groups
  - Service Group associates a group of identical devices (device type, southbound protocol, commands, etc.)
  - Service Groups are mapped to specific type of entity in the context broker and can automate device provisioning
  - Single device configuration values can be extracted from the assigned group
  - Required attributes: service group, service path, API key, type and southbound path



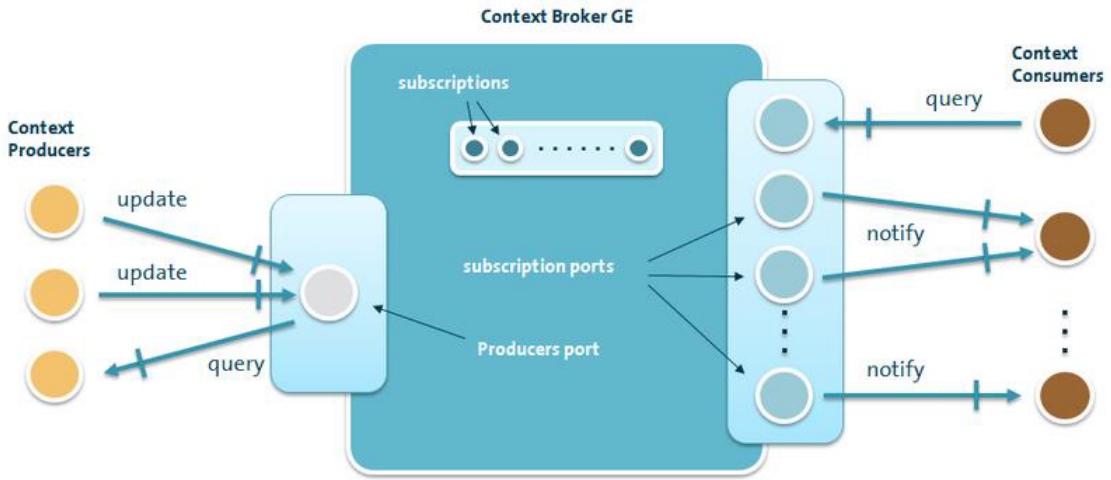
SP5
FIWARE – Context Broker

- Provides management functionalities for context information
  - Context registrations
  - Context queries
  - Context updates
  - Context subscriptions



SP5
FIWARE – Context Broker

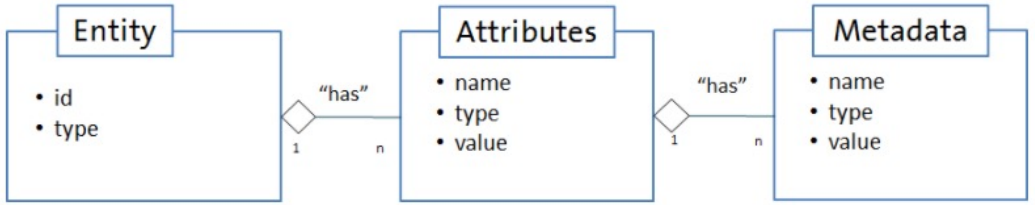
- Context registrations creates the entities for each context, allowing subsequent operations
  - Requires: context Id, context type, set of attributes and attribute types
- Context queries allows listing registered contexts or reading the state of a specific context
  - Requires: context Id
- Context updates alters the state of a specific context
  - Requires: context Id, set of attributes and respective values
- Context subscriptions offers asynchronous notifications whenever a context is updated
  - Requires: context Id, set of "monitoring" attributes, notification endpoint, set of attributes



SP5

FIWARE – Context Broker – API – Data Model

- Context Information - data model

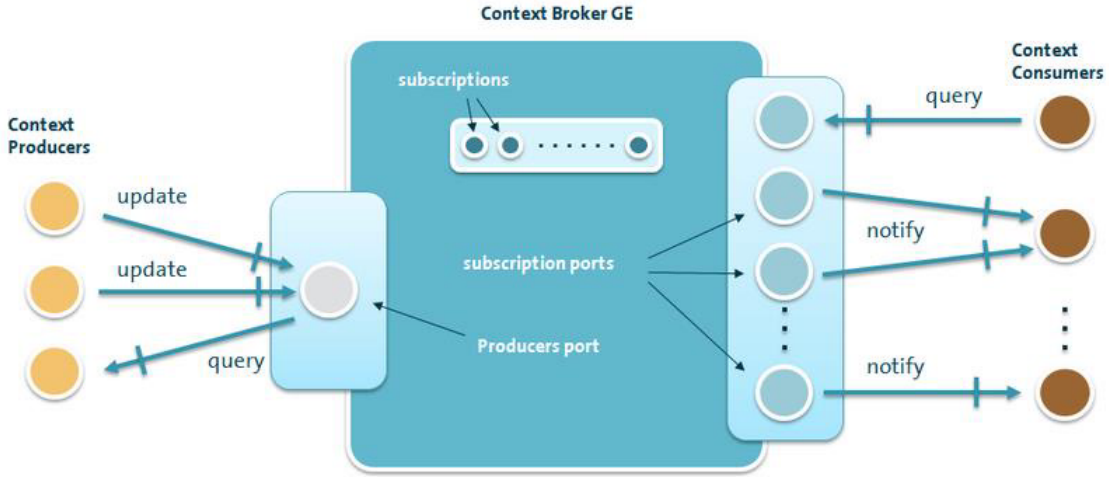


Normalized representation

```
{
  "id": "entityID",
  "type": "entityType",
  "attr_1": <val_1>,
    ...
  "attr_N": <val_N>
}
```

Example

```
{
  "id": "Room1",
  "type": "Room",
  "temperature": {
     "value": 23,
     "type": "Float" },
  "pressure": {
     "value": 720,
     "type": "Integer" }
}
```



SP5
FIWARE – Context Broker – REST API

- Entities
  - Create entity **POST** http://broker_hostname:port/v2/entities
  - List entities **GET** http://broker_hostname:port/v2/entities
  - Query entity **GET** http://broker_hostname:port/v2/entities/**id**
  - Update entity **PATCH** http://broker_hostname:port/v2/entities/**id**
  - Modify entity **POST** http://broker_hostname:port/v2/entities/**id**
  - Delete entity **DELETE** http://broker_hostname:port/v2/entities/**id**

SP5
FIWARE – Context Broker – REST API

- Entities
  - Create entity **POST** http://broker_hostname:port/v2/entities

HEADERS: Content-Type:application/json

BODY:

```
{
  "id": "Room1",
  "type": "Room",
  "temperature": {
     "value": 23.2,
     "type": "Float" },
  "pressure": {
     "value": 720,
     "type": ”Number" }
}
```

RESPONSE CODE: **201** / **204**

HEADERS: Location : /v2/entities/**Room1**?type=**Room**

SP5
FIWARE – Context Broker – REST API

- Entities
  - List entities **GET** http://broker_hostname:port/v2/entities

RESPONSE CODE: **200**
HEADERS: Content-Type:application/json
BODY:

```
[
 {
   "type": "Room",
   "id": "Room1",
   "temperature": {
     "value": 23.2,
     "type": "Float",
     "metadata": {} },
   "pressure": {
     "value": 720,
     "type": ”Number",
     "metadata": {} },
 }
]
```

SP5
FIWARE – Context Broker – REST API

- Entities
  - Query entity **GET** http://broker_hostname:port/v2/entities/**Room1**

RESPONSE CODE: **200**
HEADERS: Content-Type:application/json
BODY:

```
{
  "id": "Room1",
  "type": "Room",
  "temperature": {
     "value": 23.2,
     "type": "Float" },
  "pressure": {
     "value": 720,
     "type": ”Number" }
}
```

SP5
FIWARE – Context Broker – REST API

- Entities
  - Update entity **PATCH** http://broker_hostname:port/v2/entities/**Room1**

HEADERS: Content-Type:application/json

BODY:
```
{
  "temperature": {
     "value": 20 },
  "pressure": {
     "value": 520 }
}
```

RESPONSE CODE: **204**

SP5
FIWARE – Context Broker – REST API

- Entities
  - Modify entity **POST** http://broker_hostname:port/v2/entities/**Room1**

HEADERS: Content-Type:application/json

BODY:

```
{
  "humidity": {
     "value": 20.2,
     "type": "Float" },
  ”co2": {
     "value": 520,
     "type": "Integer" }
}
```

RESPONSE CODE: **204**

SP5
FIWARE – Context Broker – REST API

- Entities
  - Delete entity **DELETE** http://broker_hostname:port/v2/entities/**Room1**

    RESPONSE CODE: **204**

SP5
FIWARE – Context Broker – REST API

- Entity attributes
  - Query attribute **GET** http://broker_hostname:port/v2/entities/**id**/attrs/**attrName**[/value]
  - Update attribute **PUT** http://broker_hostname:port/v2/entities/**id**/attrs/**attrName**[/value]
  - Delete attribute **DELETE** http://broker_hostname:port/v2/entities/**id**/attrs/**attrName**

SP5
FIWARE – Context Broker – REST API

- Subscriptions
  - List subscription **GET** http://broker_hostname:port/v2/subscriptions
  - Create subscription **POST** http://broker_hostname:port/v2/subscriptions
  - Query subscription **GET** http://broker_hostname:port/v2/subscriptions/**id**
  - Update subscription **PATCH** http://broker_hostname:port/v2/subscriptions/**id**
  - Delete subscription **DELETE** http://broker_hostname:port/v2/subscriptions/**id**

SP5
FIWARE – Context Broker – REST API

- Subscriptions
  - Create subscription **POST** http://broker_hostname:port/v2/subscriptions

HEADERS: Content-Type:application/json

BODY:

```
{
  "description": "Description of this subscription ",
  "subject": {
        ...
    }
  },
  "notification": {
        ...
   },
  "expires": "2036-04-05T14:00:00.00Z",
  "throttling": 0
}
```

RESPONSE CODE: **201**

HEADERS: Location : /v2/subscriptions/**subscriptionId**

SP5
FIWARE – Context Broker – REST API

- Subscriptions
  - Create subscription

POST http://broker_hostnar

HEADERS: Content-Type:applicatior

BODY:

```
{
 "description": "Descripti
 "subject": {
      ...
   }
 },
 "notification": {
      ...
  },
 "expires": "2036-04-05T14:00:00.00Z",
 "throttling": 0
}
```

```
"entities": [
   {
    "idPattern" : ".*",
    "type" : "Room" }
 ],
 "condition" : {
  "attrs" : [
   "temperature" ],
  "expression" : {
   "q" : "temperature>40" }
 }
```

RESPONSE CODE: **201**

HEADERS: Location : /v2/subscriptions/**subscriptionId**

SP5
FIWARE – Context Broker – REST API

- Subscriptions
  - Create subscription

POST http://broker_hostname:port/v2/subscriptions

HEADERS: Content-Type:application/json

BODY:

```
{
 "description": "Desc
 "subject": {
      ...
   }
 },
 "notification": {
      ...
 },
 "expires": "2036-04-05T14:00:00.00Z",
 "throttling": 0
}
```

```
"http": {
   "url": "http://notification_end_point:port"
 },
 "attrs": [
  ”humidity",
  "temperature"
 ]
```

RESPONSE CODE: **201**

HEADERS: Location : /v2/subscriptions/**subscriptionId**

SP5
FIWARE – Context Broker Notifications

- Notifications

POST http://notification_end_point:port
HEADERS:
Content-Type: application/json
...
BODY:

```
{
  "data": [
    {
      "id": "Room1",
      "temperature": {
        "metadata": {},
        "type": "Float",
        "value": 28.5
      },
      "type": "Room"
    }
  ],
  "subscriptionId": "57458eb60962ef754e7c0998"
}
```

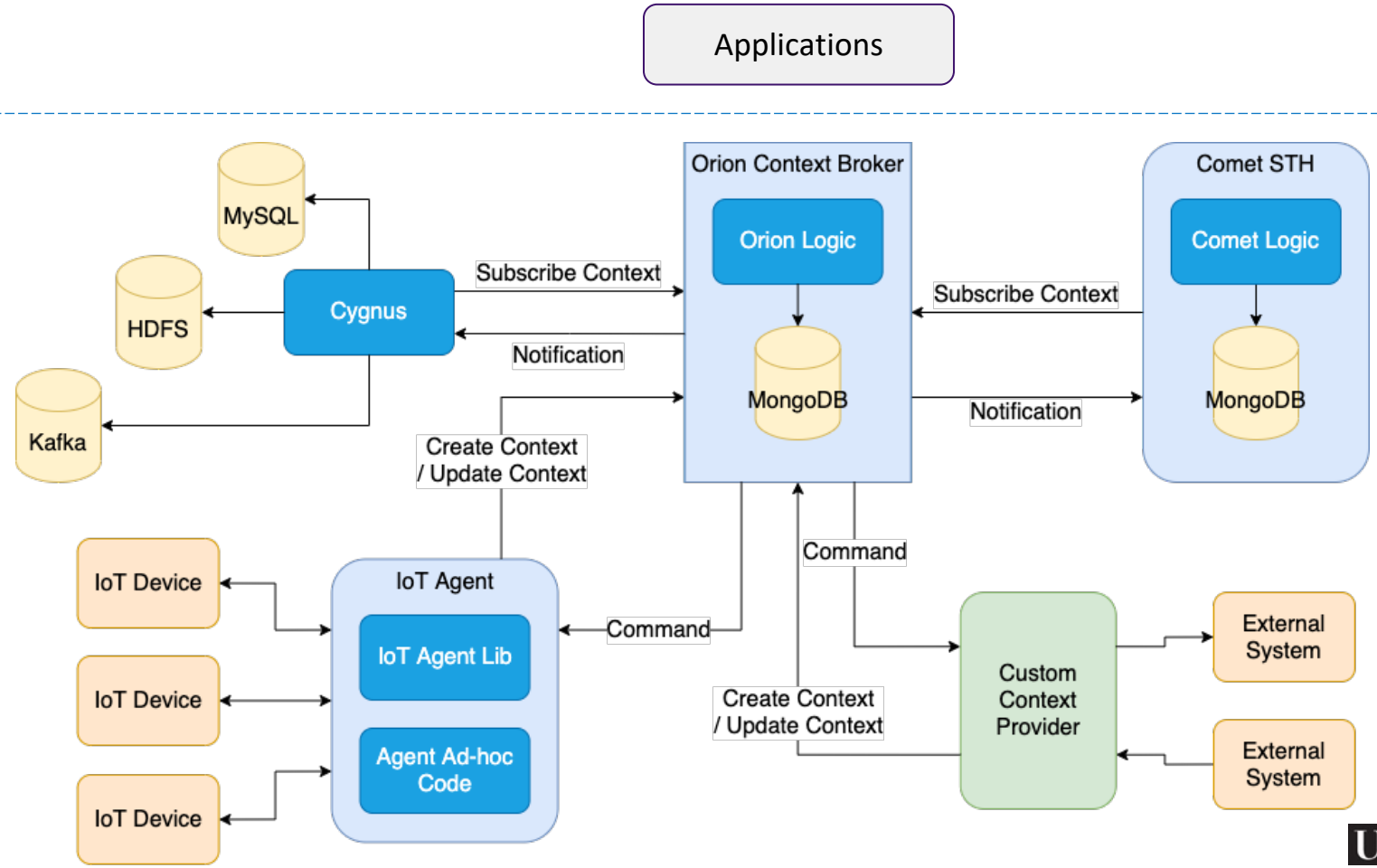
FIWARE ARCHITECTURE & MODELS

Safe Cities
Building Urban Safety

U.PORTO
BOSCH

SP5

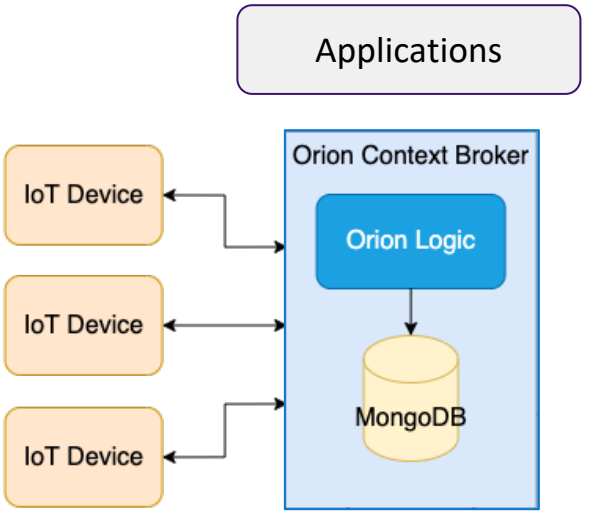
FIWARE – Generic Architecture



SP5
FIWARE – Architecture variations

- Modular approach allows developers to chose the components they require to build their applications
  - The only obligatory component is the Context Broker

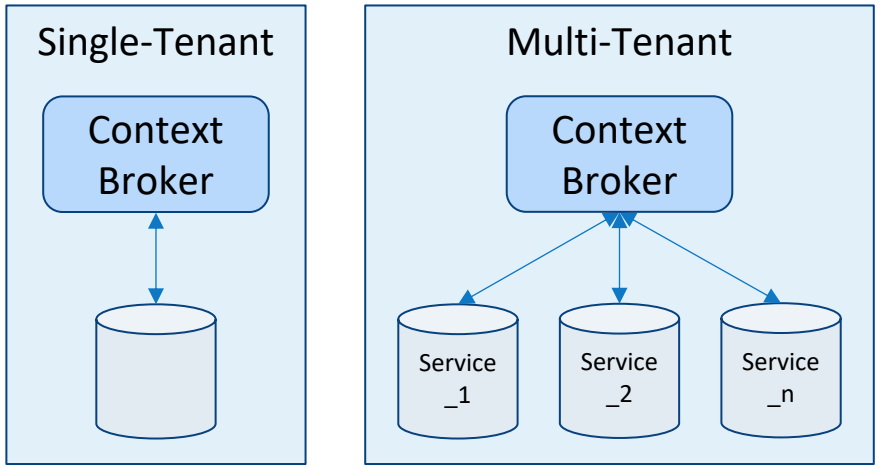
- Minimal architecture / deployment
  - Devices “connect” directly to the Context Broker
  - Applications consume context information directly from the context broker



SP5
FIWARE – Single-Tenant vs Multi-Tenancy

- Single-tenant configurations
  - All devices and applications share the same broker
    - A single DB instance is created
  - There is no separation between Contexts information
  - Applications can access all registered entities, Contexts information, subscriptions, etc.

- Multi-tenant configurations
  - Both devices and applications share the same broker
    - Each tenant has an isolated DB instance
    - DB instances are started on the fly
  - Context information is stored based on the Service path
    - Given by the Fiware-Service header



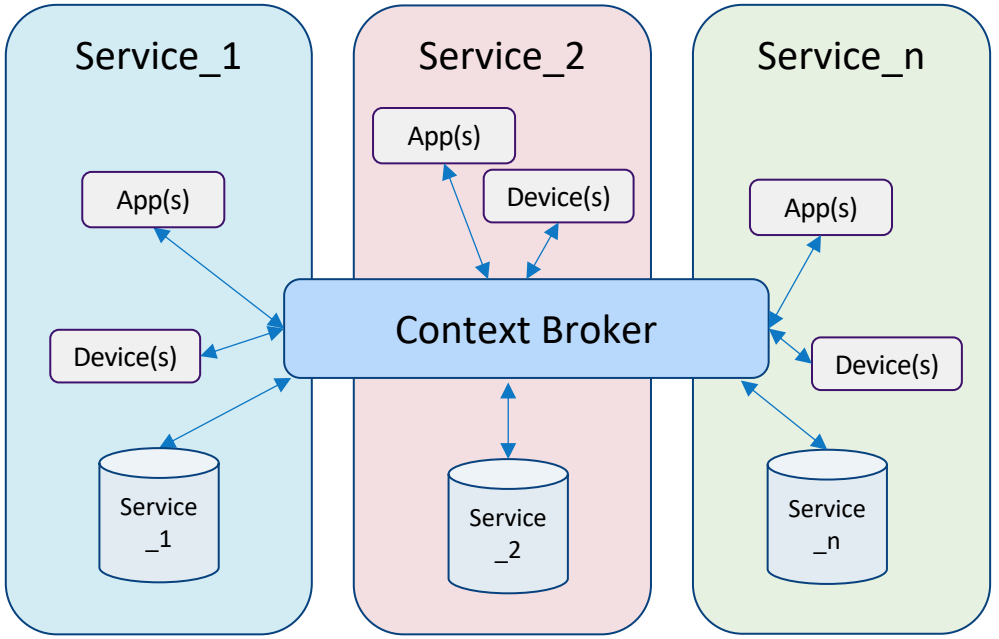
SP5
FIWARE – Multi-Tenant Model

- Multi-tenant configurations
  - Create entity

POST http://broker_hostname:port/v2/entities
HEADERS: Content-Type: application/json
Fiware-Service: service_path

BODY: {
. . .
}

RESPONSE CODE: **201** / **204**
HEADERS: Location : /v2/entities/**id**?type=**type**
Fiware-Service: **service_path**



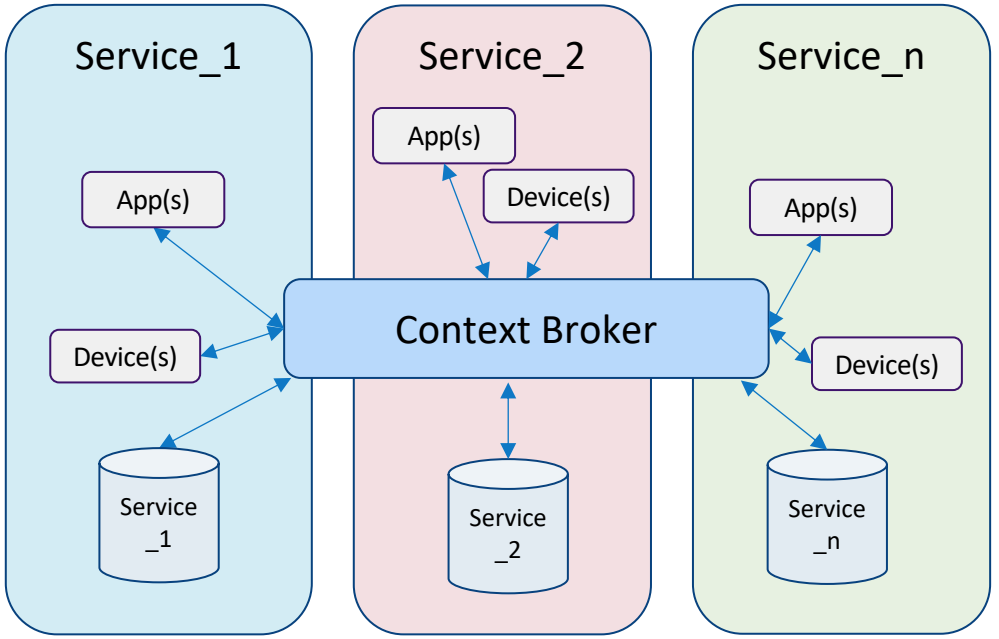
SP5
FIWARE – Multi-Tenant Model

- Multi-tenant configurations
  - Update entity

PATCH http://broker_hostname:port/v2/entities/**Room1**
HEADERS: Content-Type: application/json
Fiware-Service: service_path

BODY: {
. . .
}

RESPONSE CODE: **204**
HEADERS: Fiware-Service: **service_path**



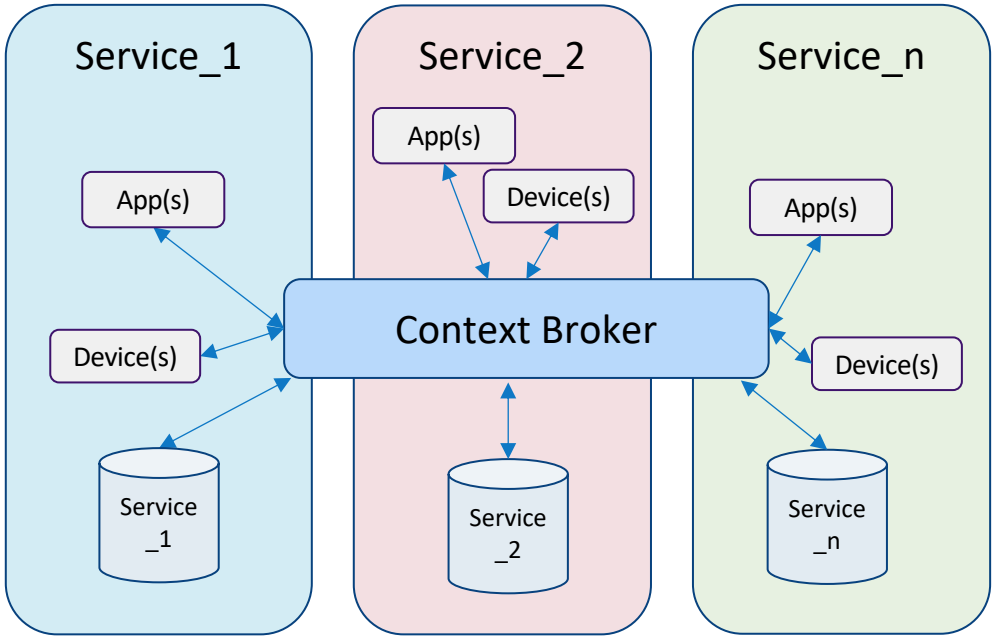
SP5
FIWARE – Multi-Tenant Model

- Multi-tenant configurations
  - Create subscription

POST http://broker_hostname:port/v2/subscriptions
HEADERS: Content-Type: application/json
Fiware-Service: service_path

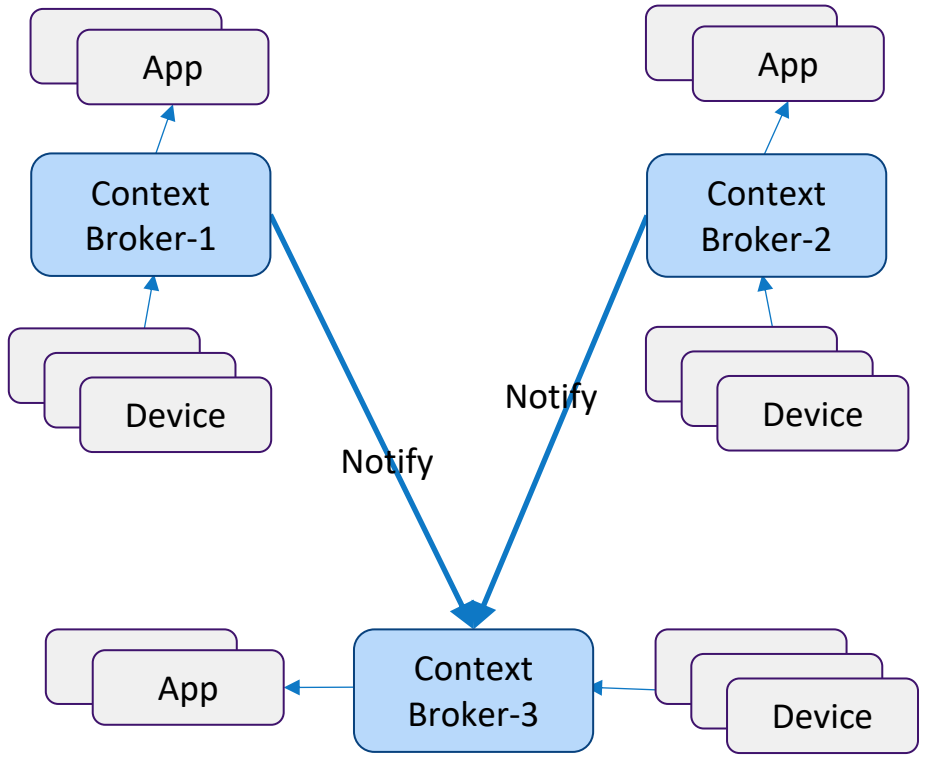
BODY: {
. . .
}

RESPONSE CODE: **201**
HEADERS: Location : /v2/subscriptions/**subscriptionId**
Fiware-Service: **service_path**



SP5
FIWARE – Federated Architecture

- Federated Architecture allows Context Brokers to push notifications to other Brokers
  - All involved brokers store the ”pushed” value and notify their respective clients
- Federation allows cooperation between deployment locations, providers, etc.
- Federation does not provide mirroring
  - If an entity is deleted in Context Broker A the entity will not be deleted in any other Federated Broker.



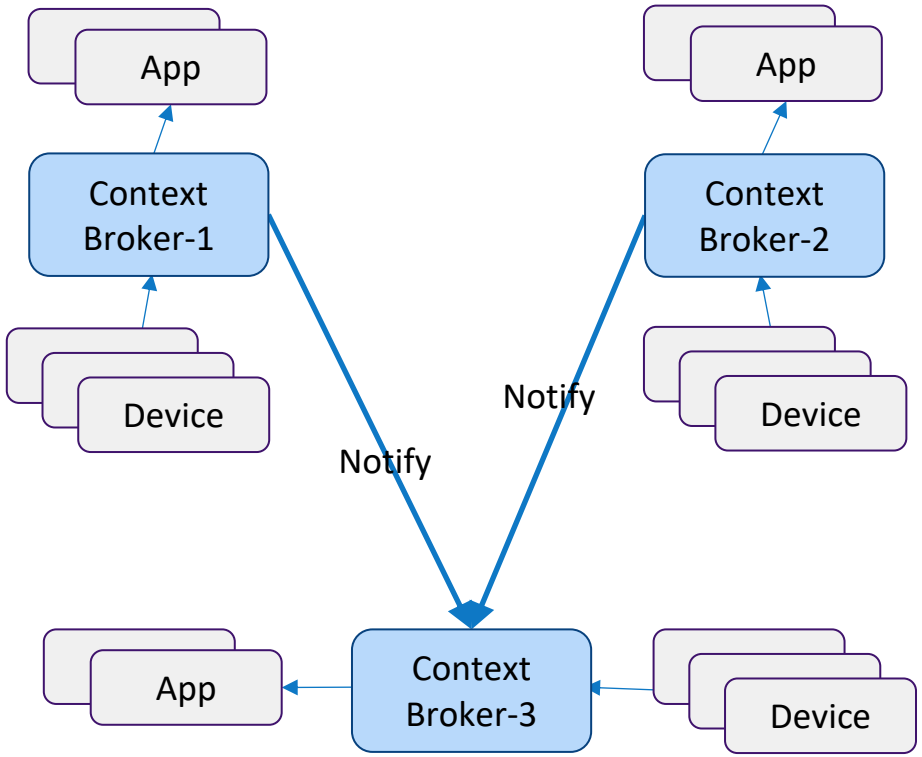
SP5
FIWARE – Federated Architecture

- Context Brokers subscribe to other brokers in the usual way
- The Notification end-point has to be /v2/op/notify

POST http://broker_hostname:port/v2/subscriptions
HEADERS: Content-Type: application/json
Fiware-Service: service_path

BODY:
```
{
  . . .
   "http": {
      "url": "http://broker_host:port/v2/op/notify"
   },
  . . .
}
```

RESPONSE CODE: 201
HEADERS: Location : /v2/subscriptions/subscriptionId
Fiware-Service: service_path



SP5

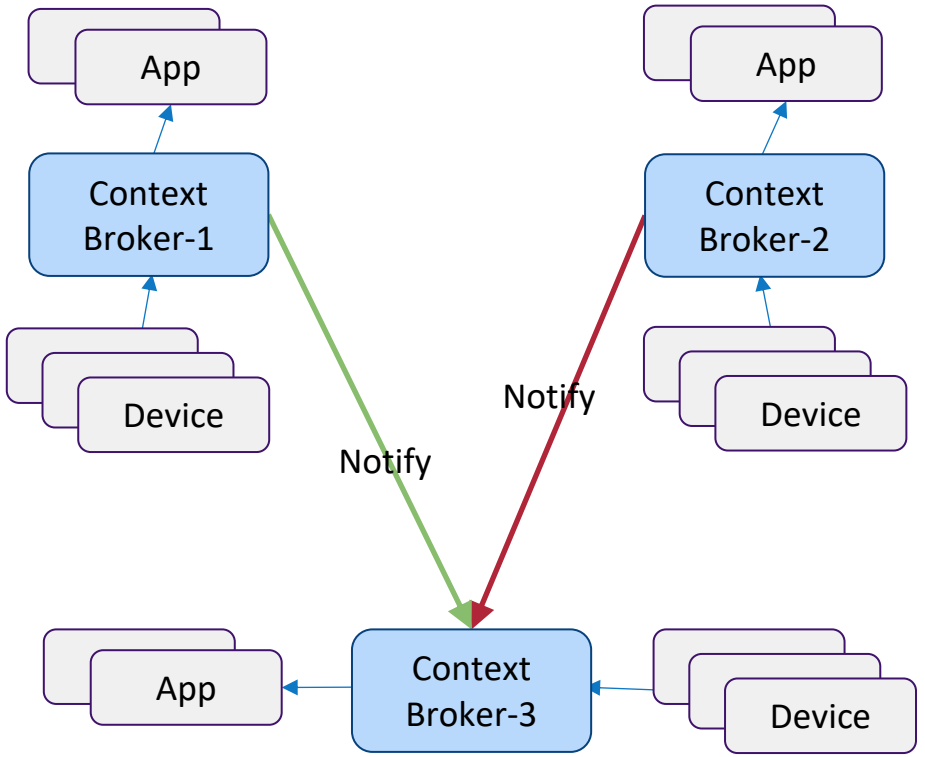
FIWARE – Federated Architecture

POST http://**broker_1**/v2/subscriptions
HEADERS: Content-Type: application/json
Fiware-Service: service_path

BODY:
```
{ . . .
  "http": {
    "url": "http://broker_3/v2/op/notify"
  },
 . . . }
```

POST http://**broker_2**/v2/subscriptions
HEADERS: Content-Type: application/json
Fiware-Service: service_path

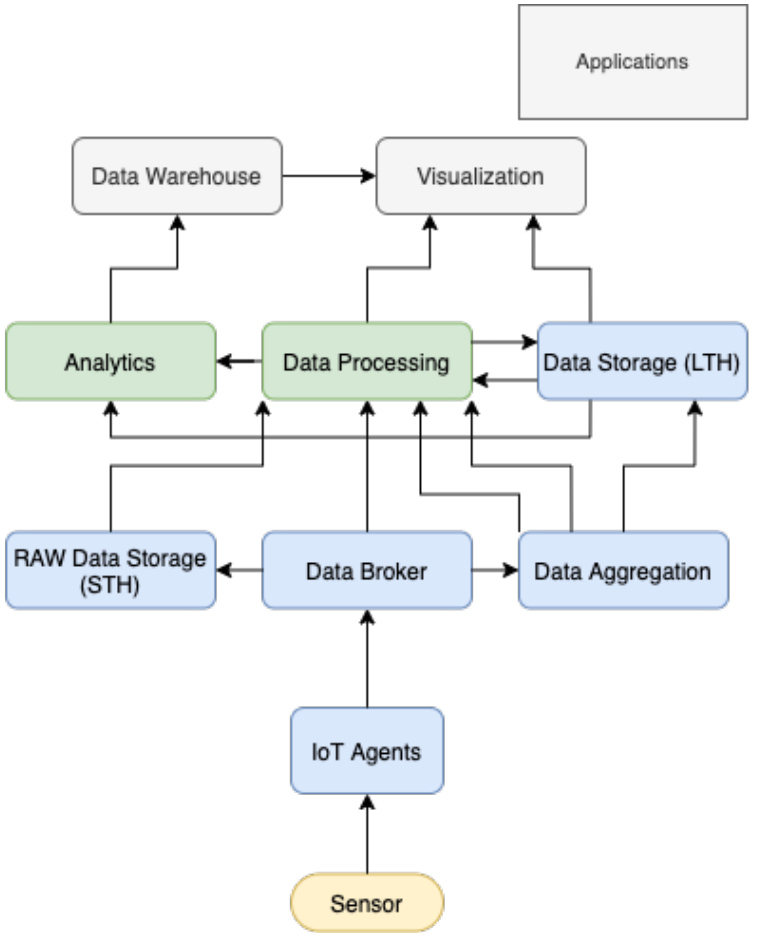
BODY:
```
{ . . .
  "http": {
    "url": "http://broker_3/v2/op/notify"
  },
 . . . }
```



SP5
FIWARE – Deployment (Local)

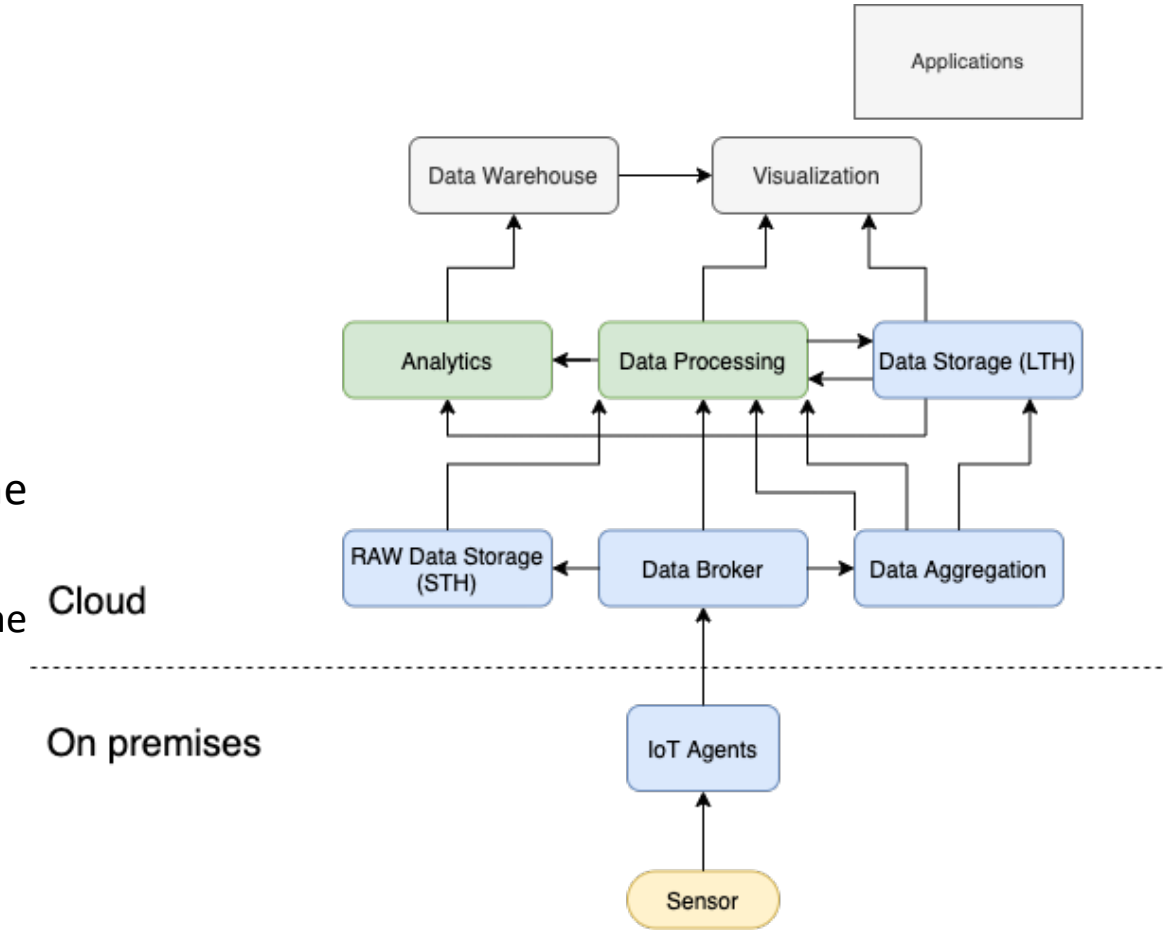
- All components are deployed “on premises”
- Data is acquired, stored, processed, analysed, and displayed “on premises”
- Applications run “on premisses”

On premises



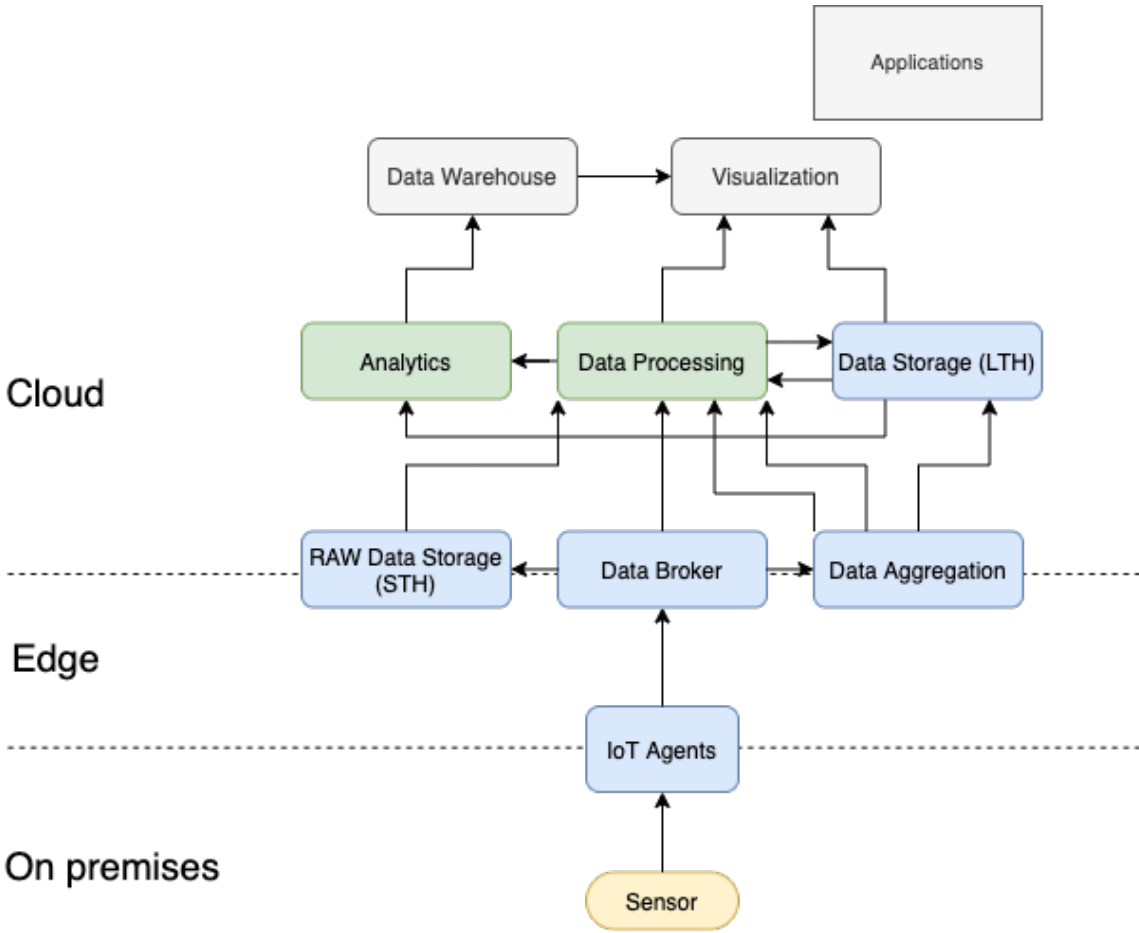
SP5
FIWARE – Deployment (Cloud)

- Devices and Agents are deployed “on premises”
  - Data is acquired “on premises”
- Data Management components are deployed on the Cloud
  - Data is stored, processed, analysed, and displayed on the Cloud
- Applications run on the Cloud



SP5
FIWARE – Deployment (Hybrid)

- Devices are deployed “on premises”
  - Data is acquired “on premises”
- Agent can be deployed “on premises” and on the Edge
- Data Management components are deployed on the Edge or on the Cloud
  - STH Data is stored, processed and aggregated on the Edge
  - LTH Data is stored, processed, analysed, and displayed on the Cloud
- Applications run on the Cloud



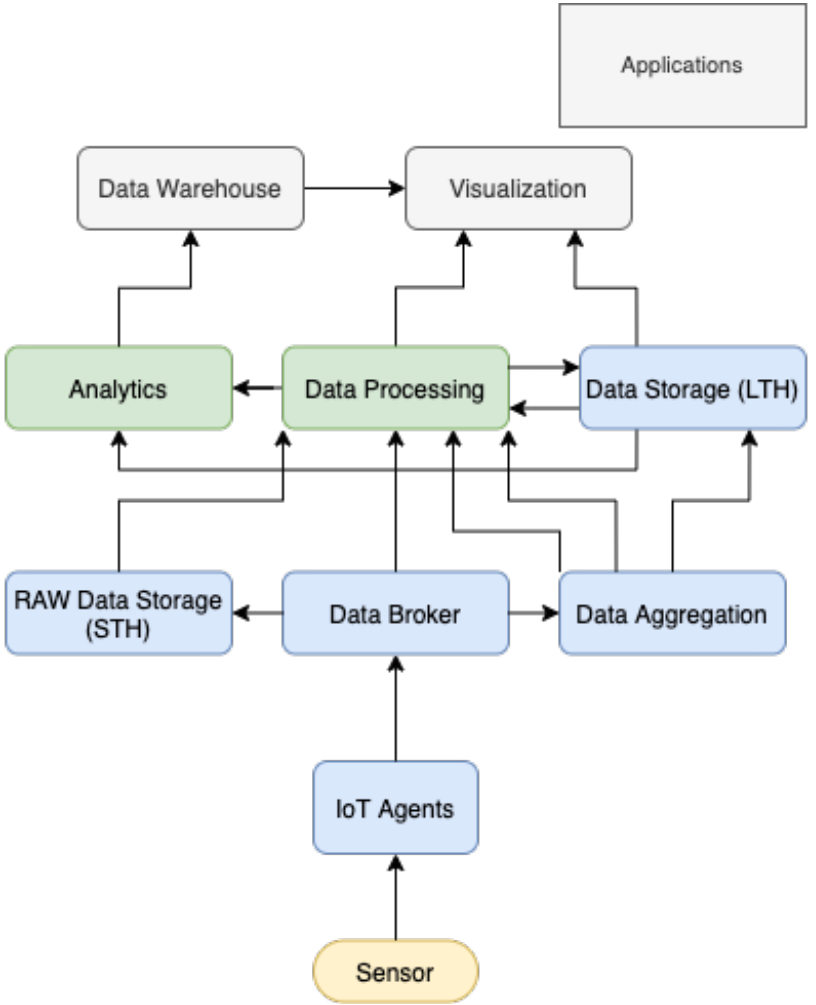
FIWARE APPLICATIONS

Safe Cities
Building Urban Safety

U.PORTO
BOSCH

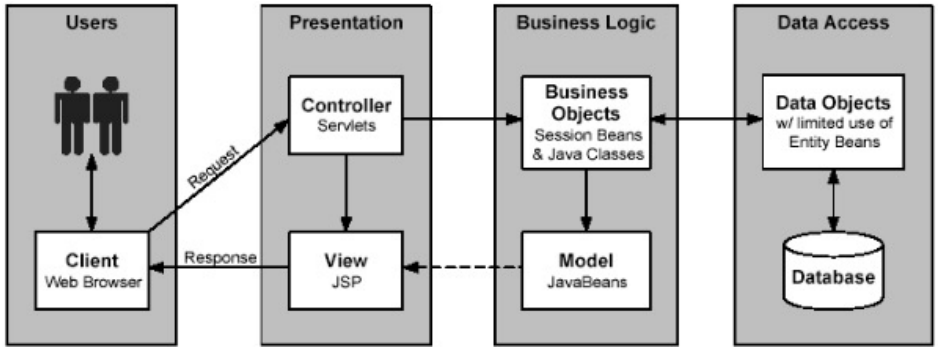
SP5
FIWARE – Application Architecture

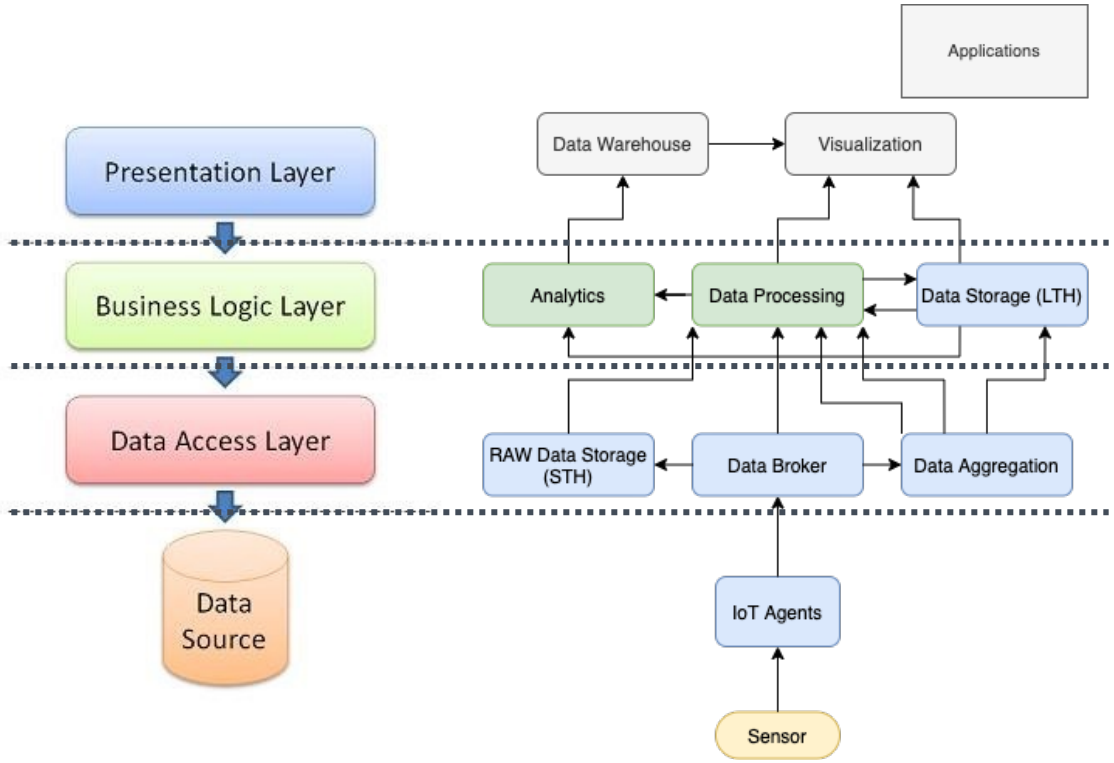
- IoT applications typically
  - Acquire data from IoT devices
  - Store the acquired (raw) data
  - Process and transform raw data into information
    - According to the respective application
  - Store processed information and/or execute procedures based on this information
  - Display the processed information, reports, etc.



SP5
FIWARE – Application Architecture

- IoT applications typically follow similar design and architecture principles of Web Application
  - n-Tier Architecture / Model – View – Controller
    - Data Source -> IoT devices; IoT Agents; etc.
    - Data Management -> Broker; STH; LTH; etc.
    - Business Logic -> Data Processing; Analytics; etc.
    - Presentation -> . . .
  - Based on Micro-Services
    - Each layer is self contained with REST API





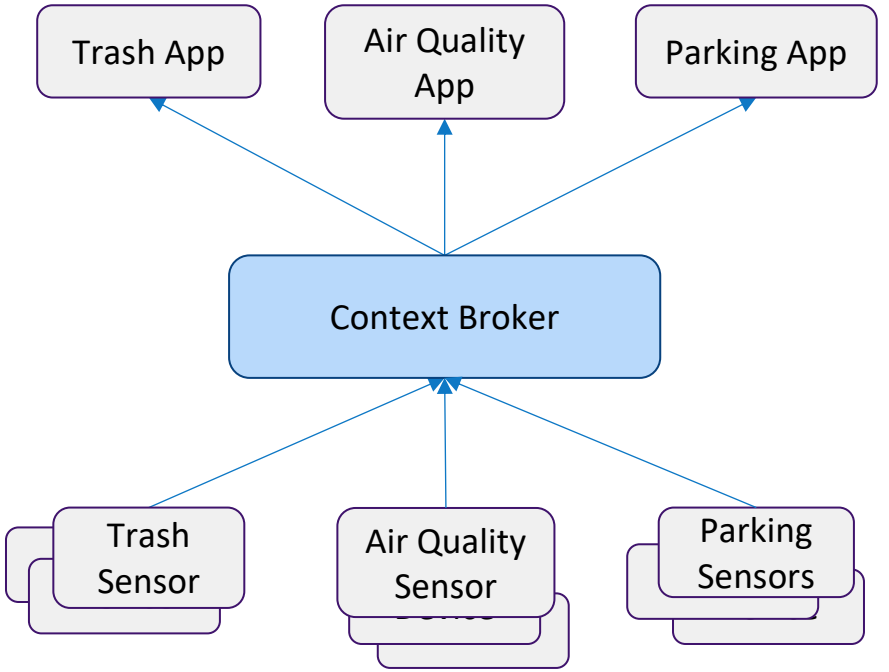
DEMO APPLICATIONS

Safe Cities
Building Urban Safety

U.PORTO
BOSCH

SP5
FIWARE – Demo Applications

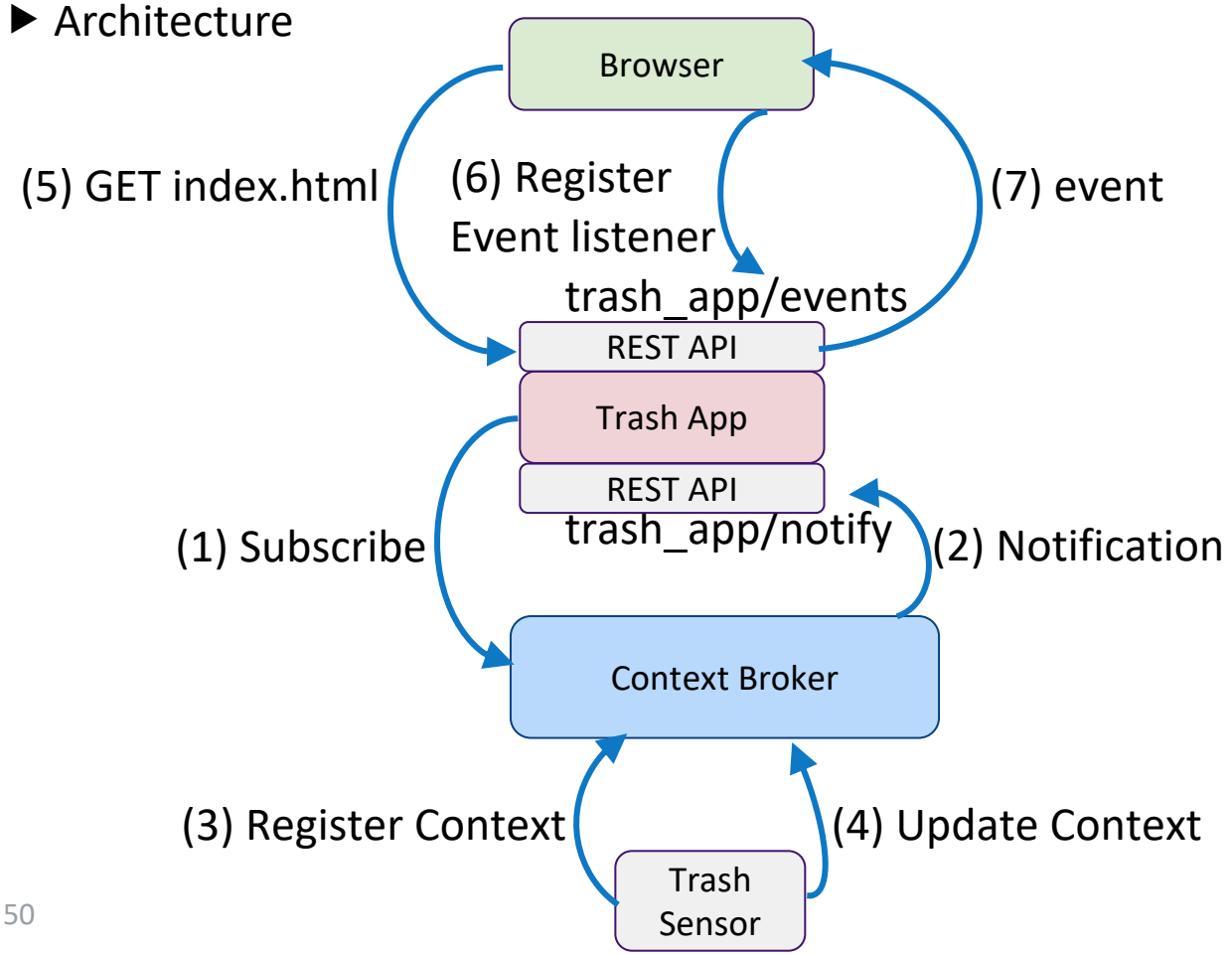
- Waste Management Application
  - Monitors level of waste containers and displays them over a map
- Air Quality Monitoring Application
  - Monitors air quality sensors and displays them over a map
- Parking Management Application
  - Monitors occupancy level of parking infrastructures and displays them over a map

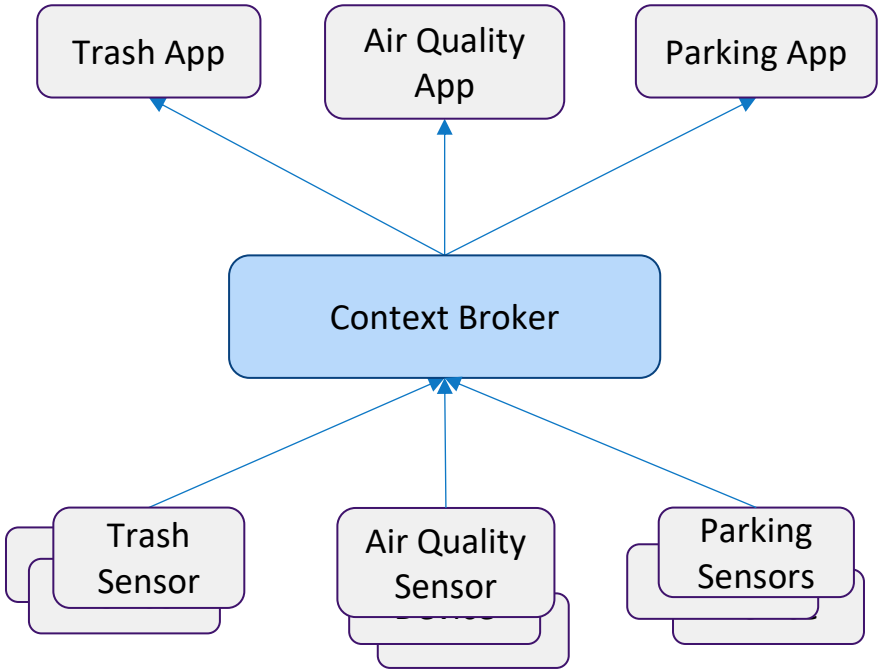


SP5

FIWARE – Demo Applications - Architecture

- Architecture

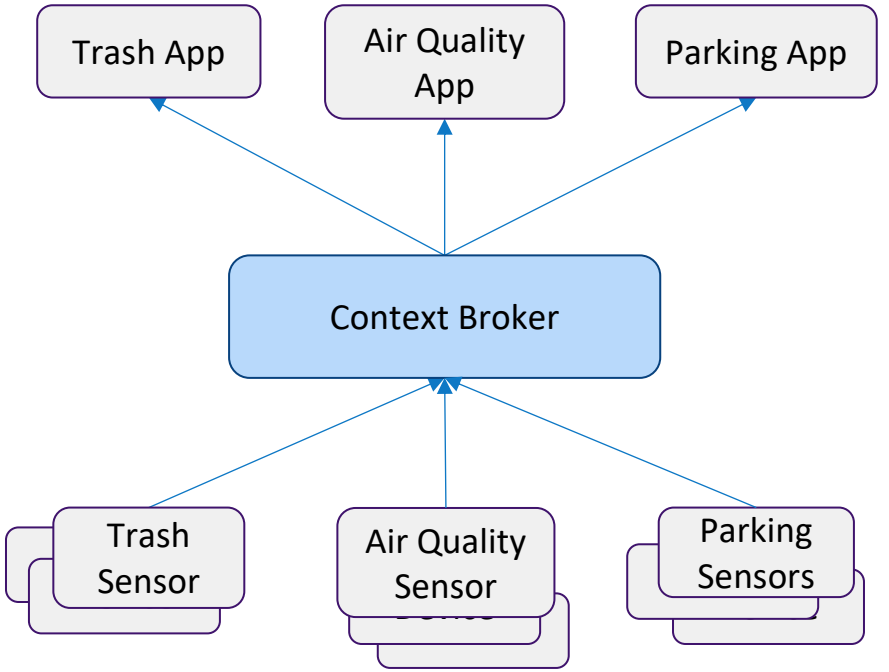




SP5
FIWARE – Demo Application – Single-Tenant

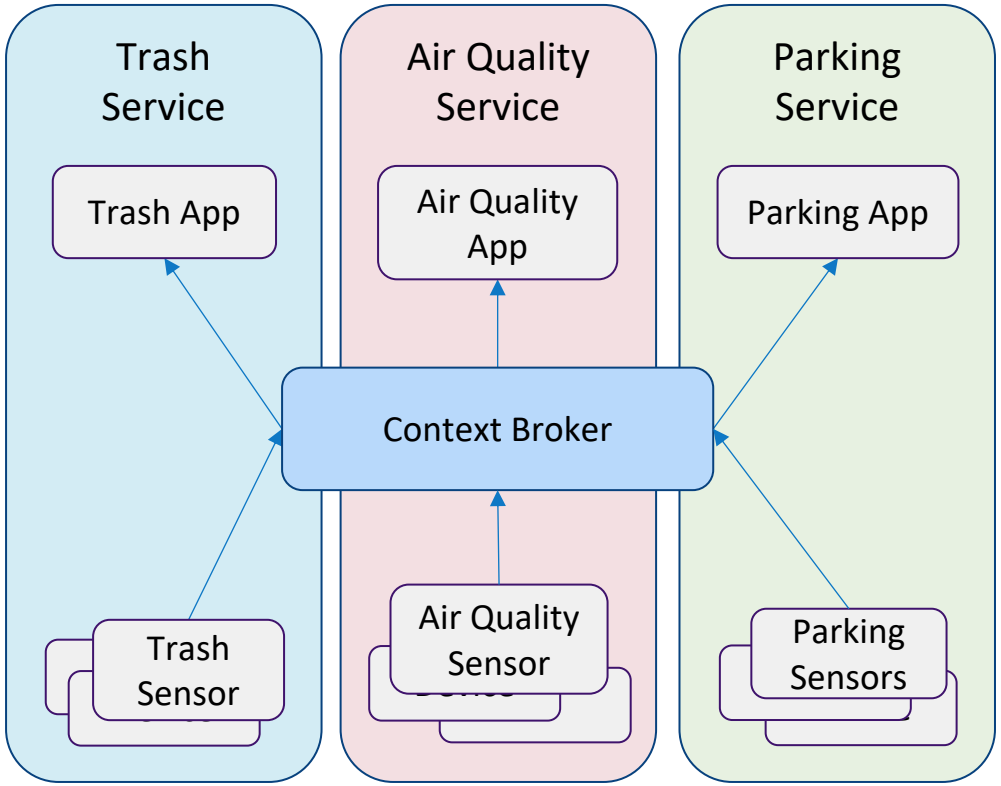
- All sensors and sensor types publish content information to a single Broker
  - Using the default service
- Apps subscribe their respective context information from the same broker

- No context information separation is performed
  - **All** Apps have access to **all entities, context information, etc.**



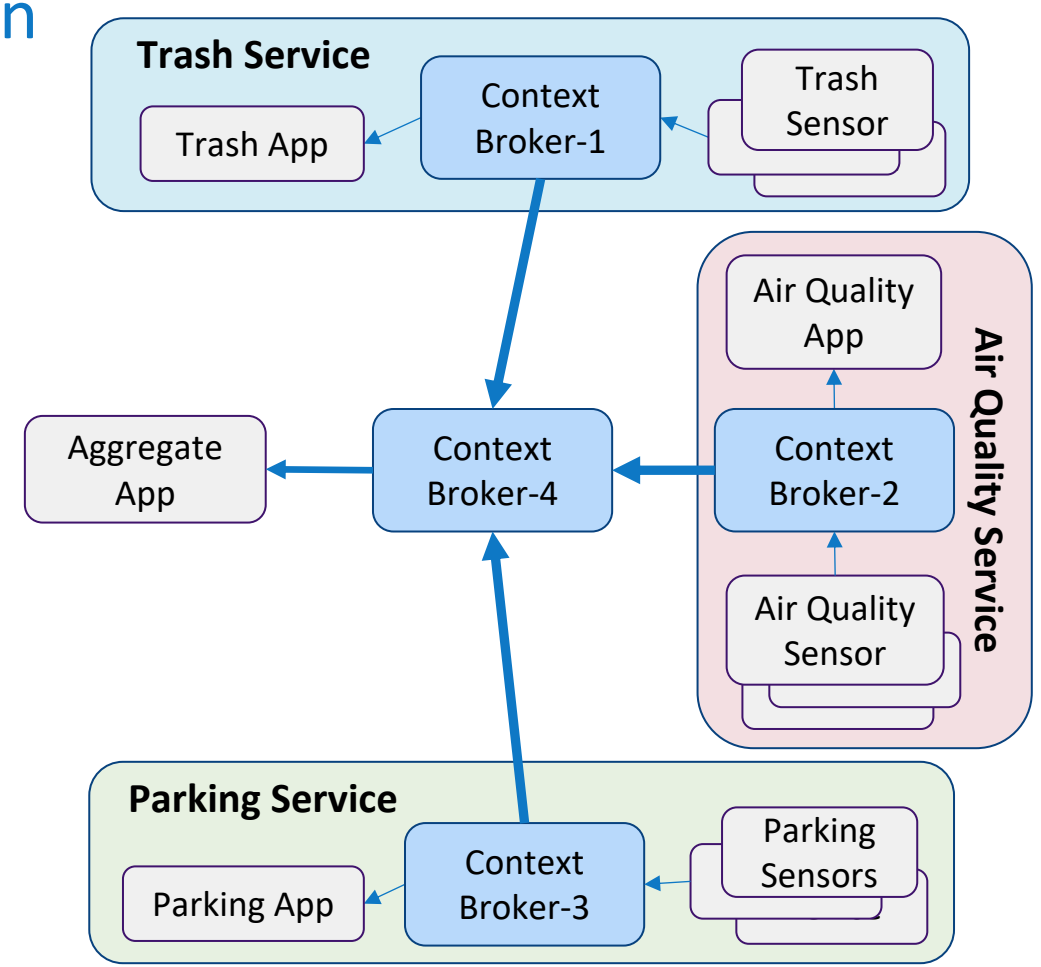
SP5
FIWARE – Demo Application – Multi-Tenant

- All sensors publish content information using a Service path based on the sensor type
  - Each sensor type is associated to its own service
- Apps subscribe to context information based on the respective sensor type and Service path
- Context information is physically separated by the Service path
  - Apps only have access to the **Service's entities, context information, etc.**



SP5
FIWARE – Demo Federated Application

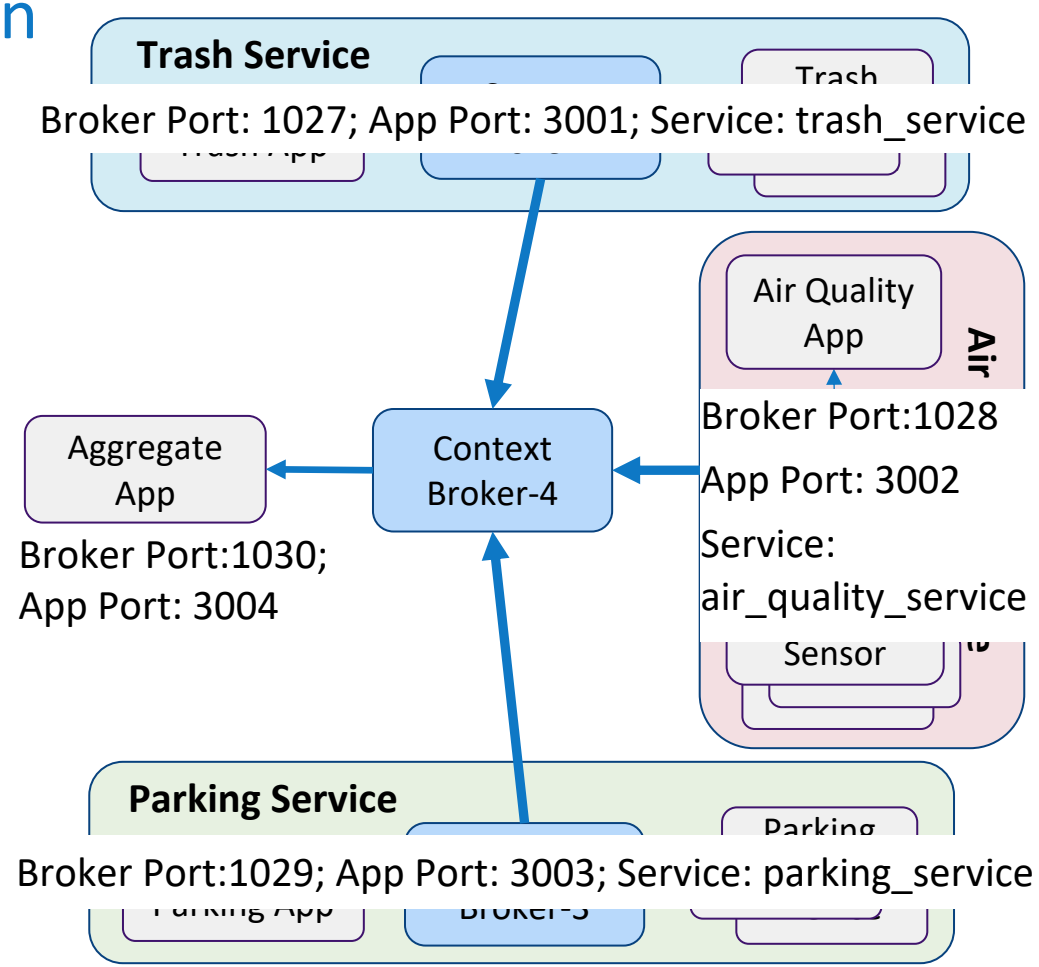
- **Federation is composed of 4 participants**
  - Broker_1, Broker_2, Broker_3, Broker_4
- Each single sensor type and respective app are mapped to an individual Broker and Service path
- Broker_4 subscribes to the other Brokers for their respective Context information
- Broker_4 “serves” the application that aggregates Context information from all sensors of all types



SP5
FIWARE – Demo Federated Application

- **Federation is composed of 4 participants**
  - Broker_1, Broker_2, Broker_3, Broker_4
- Each single sensor type and respective app are mapped to an individual Broker and Service path

- Broker_4 subscribes to the other Brokers for their respective Context information
- Broker_4 "serves" the application that aggregates Context information from all sensors of all types



THANK YOU

Safe Cities
Building Urban Safety

U.PORTO BOSCH

Workshop
FIWARE

25 February
9h – 13h

Cofinanciado por:
COMPETE 2020
PORTUGAL 2020
UNIÃO EUROPEIA
Fundo Europeu
de Desenvolvimento Regional

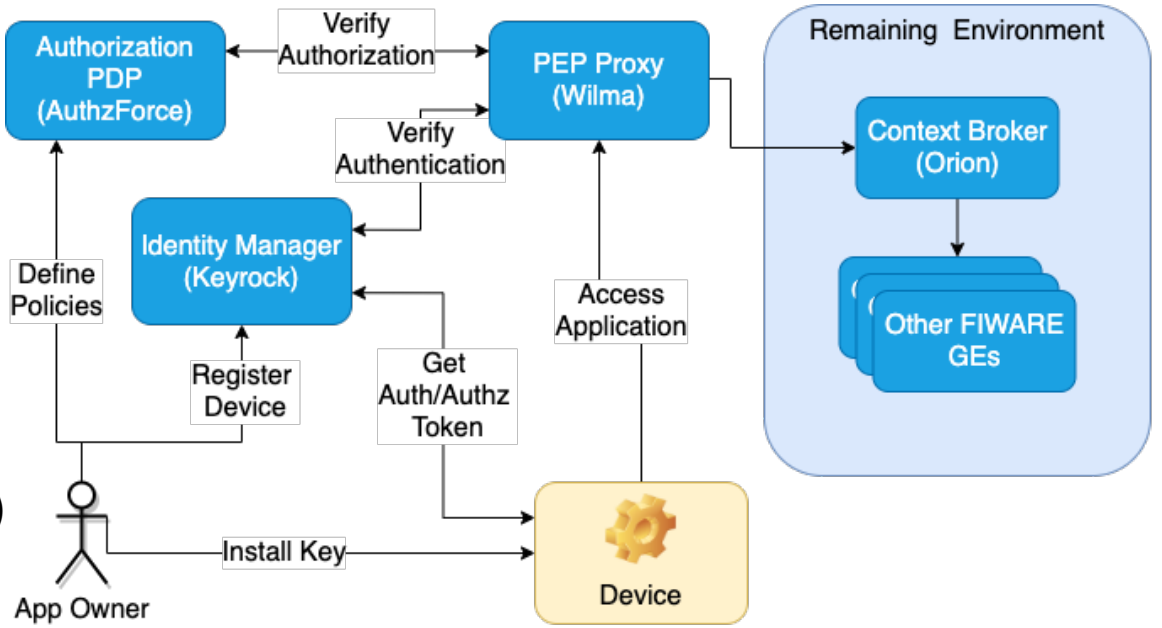
SECURITY

Safe Cities
Building Urban Safety

U.PORTO
BOSCH

SP5
FIWARE – Security

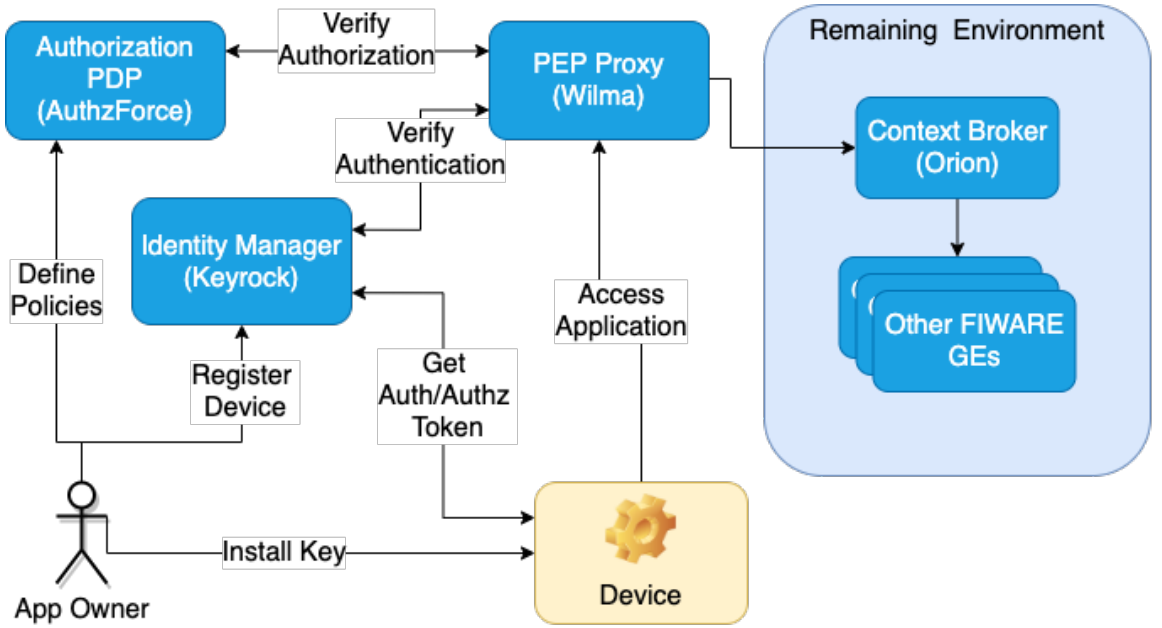
- FIWARE follows a modular approach to Security
- Providing **Authentication** and **Authorization** components
  - **KeyRock Identity Manager (IdM)** managing devices, users and applications
  - **AuthzForce** – reference implementation of PDP (Policy Decision Point) and the PAP (Policy Administration Point)
  - **Wilma** – reference implementation of a Policy Enforcement Point (PEP)



SP5
FIWARE – Identity Management GE

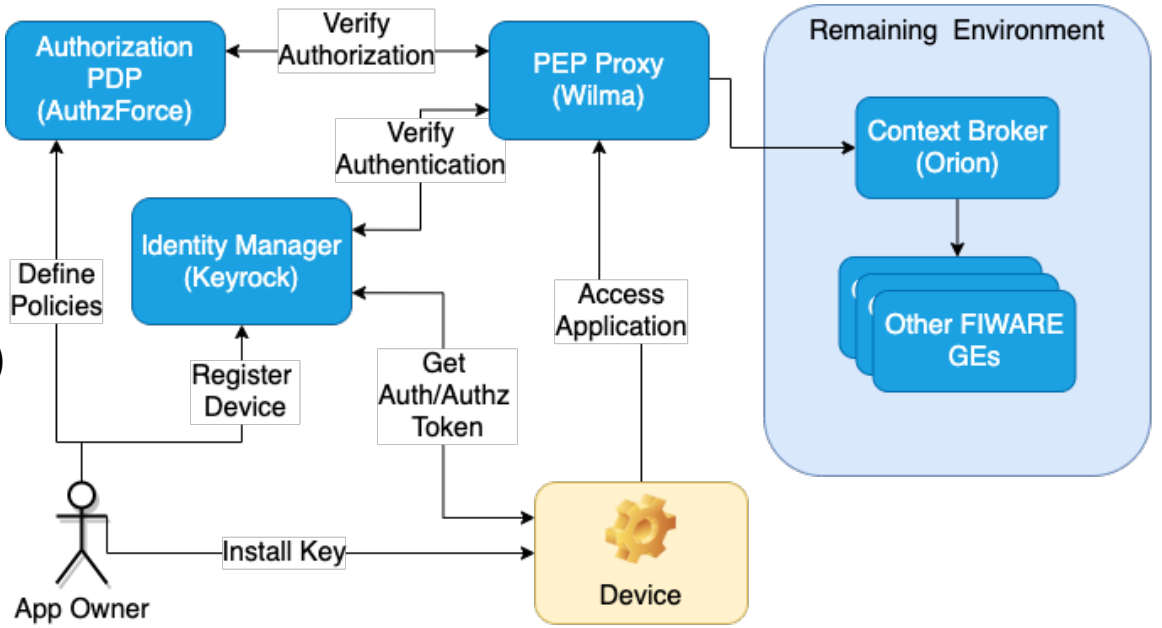
- Identity Management authorizes foreign services to access data stored in a secure environment

- Bridges IdM at connectivity- and application-level
  - Secure and private authentication for users, devices, networks and services
  - Authorization & trust management
  - User profile management, and privacy-preserving disposition of personal data
  - Single Sign-On (SSO) to service domains
  - Identity Federation towards applications



SP5
FIWARE – Security

- FIWARE provides **Authentication** and **Authorization** components
  - **KeyRock Identity Manager (IdM)** managing devices, users and applications
  - **WILMA** – reference implementation of a Policy Enforcement Point (PEP)
  - **AuthZForce** – reference implementation of PDP (Policy Decision Point) and the PAP (Policy Administration Point)

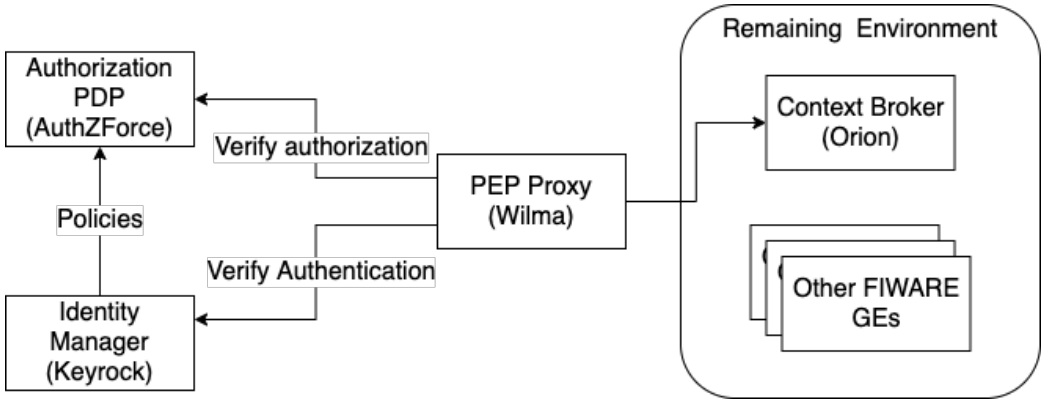


- **WILMA** is a proxy-based PEP
  - Intercepts requests and Authenticates and Authorizes them contacting the IdM and PDP

- Based on a modular approach and **not "by design"**

SP5
FIWARE – Security

- FIWARE provides **Authentication** and **Authorization** components
  - **KeyRock Identity Manager (IdM)** managing devices, users and applications
  - **WILMA** – reference implementation of a Policy Enforcement Point (PEP)
  - **AuthZForce** – reference implementation of PDP (Policy Decision Point) and the PAP (Policy Administration Point)



- **WILMA** is a proxy-based PEP
  - Intercepts requests and Authenticates and Authorizes them contacting the IdM and PDP

- Based on a modular approach and **not "by design"**