Be smart in a SNAP!

LIVING LAB

Smart IOT Applications & IOT Networks

19 January 2022, Course
https://www.snap4city.org/577
scalable Smart aNalytic APplication builder for sentient Cities: for Living Lab and co-working with Stakeholders

https://www.Snap4City.org

Smart IOT Applications & IOT Networks

19 January 2022, Course
https://www.snap4city.org/577

Paolo Nesi, paolo.nesi@unifi.it
https://www.Km4City.org
https://www.disit.org
Tools for rapid implementation of sustainable Smart Solutions and Decision Support Systems

Dashboards and Apps - Control Rooms - Decision Support Systems - What-if Analysis - Visual Analytics

Prediction - Anomaly Detection - Environmental Model - 3D Model
KPI - Simulation - Early Warning - Synoptic - Digital Twin - Virtual Reality

Expert System
Knowledge Base
Storage

Big Data Analytics
Explainable Artificial Intelligence
Business Intelligence
Machine Learning

Data Flows, Data Driven Workflows, Microservices
Parallel Distributed Processing

Methodologies
Courses and Community
Living Labs
Development Tools

Industry 4.0
Environment
Health
Security
Satellite

Parking
Agriculture
Energy
Mobile
Waste
Building
Transport

E015
digital ecosystem

Snap4City (C), January 2022
Cloud vs Fog/Edge Computing

Cloud Layer

Edge/Fog Layer

Smart Devices

IoT HARDWARE Technology Guide

IoT LONG RANGE DEVICES

IoT SMARTPHONE + DEVICES

PAN

LAN

WAN
Snap4City/Industry structure

• The Snap4xxxx solution is released in Open Source, VM and Docker with fully support of MultiTenant/multiple-O rganizations
  – Each Organization may be configured for a separate environment with a set of Maps, Menus, Users, Data, Dashboards, IOT Apps, MicroApplications, Custom Widgets, Models, resources, open data, etc.

•Https://www.Snap4City.ORG is the main instance of Snap4xxxx solution managed by DISIT Lab. The main documentation is located and updated on Snap4City.org, GitHub, dockerHub and Node-Red Library. Snap4City.org is where the last tools are tested and news published.
  – Organizations on Snap4City.org have been created with contracts as for Platform as a Service, for testing and for providing SmartCity as a Service as well as Industry 4.0 as a Service

Snap4City (C), January 2022
• Most of Organizations on Snap4City.org also correspond to companies or institutions that have an installation of Snap4City tools on their Premise,
  – such as: Pisa, SmartGarda Lake, Snap4, ALTAIR, etc.
• This double way allows them to:
  – test the news,
  – share experiences with other groups,
  – get visibility,
  – work in the collaborative environment, and
  – be better supported by Snap4City.org and DISIT Lab personnel.
• Each instance of Snap4xxxx solution \textit{can decide to join the federation} of Smart City API to exploit shared data.
  – This allows to exploit regional data for city installations applications (web, mobile, dashboards, etc.) without reloading them for example.
Free Trial

- Register on [WWW.snap4city.org](http://WWW.snap4city.org)
  - Subscribe on DISIT Organization
- You can:
  - Access on basic Tools
  - Access to a large volume of Data
  - Create Dashboards
  - Create IOT Applications
  - Connect your IOT Devices
  - Exploit Tutorials and Demonstrations

IF you need to go more in deep you can ask us to pass at the next Role becoming full AreaManager with full rights of development, also for Data Analytics, machine learning, etc.
**On Line Training Material** (free of charge)

<table>
<thead>
<tr>
<th>what</th>
<th>1st part (*)</th>
<th>2nd part (*)</th>
<th>3rd part (*)</th>
<th>4th part (*)</th>
<th>5th part (*)</th>
<th>6th part (*)</th>
<th>7th part (*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF</td>
<td>General</td>
<td>Dashboards</td>
<td>IOT App, IOT Network</td>
<td>Data Analytics</td>
<td>Data Ingestion processes</td>
<td>System and Deploy Install</td>
<td>Smart City API: Web &amp; Mob. App</td>
</tr>
<tr>
<td>Interactive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video1</td>
<td>YouTube</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video2</td>
<td>YouTube</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video3</td>
<td>YouTube</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video4</td>
<td>YouTube</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>duration</td>
<td>2:55</td>
<td>3:16</td>
<td>3:41</td>
<td>2:00</td>
<td>2:48</td>
<td>2:35</td>
<td>1:47</td>
</tr>
</tbody>
</table>

[https://www.snap4city.org/577](https://www.snap4city.org/577)
General Overview of the full Course 2021

• **1st part:** General Overview
• **2nd part:** Dashboards Creation and Management
• **3rd part:** IOT Applications development, IOT Devices, IOT Networks
• **4th part:** Data Analytics, in R Studio, in Python, how to Exploit and Manage Data Analytics in IOT Applications
• **5th part:** Data Ingestion, Data Warehouse, Data Gate, IOT Device Data ingestion, IOT App for Data Ingestion, Interoperability, etc.
• **6th part:** Snap4City Development, Extension, Administration, and Installation
• **7th part:** Smart city API (internal and external) Web and Mobile App development tool kit

A number of the training sections include exercitations

Updated versions on: [https://www.snap4city.org/577](https://www.snap4city.org/577)
See also courses in ITALIANO: [https://www.snap4city.org/485](https://www.snap4city.org/485)
IOT App Smartening Dashboards and Solutions
- IOT App Smartening Devices and Dashboards, Smart Parking, IOT App vs Smart City Solutions
- IOT App Smart Industry 4.0 Snap4Industry, ModBus Integration
- IOT App vs Smart Home Snap4Home, Moving IOT Devices / Sensors, Tracking Devices

Managing IOT Applications and Containers all
- Remote control of IOT Applications on IOT edge devices, and Containers all

Creating IOT Applications with Node-RED
- Node-RED Hello World

IOT App = Node-RED + Snap4City
- IOT App = Node-RED + Snap4City solution, demo, Synoptics, NGSI
- IOT App self training, IOT App examples, IOT App and Dashboard Integration, dynamic widgets, IOT App vs Synoptics in real time data driven

Data Analytics and web Scraping
- IOT App vs Data Analytics: R-Studio, Python; IOT App and web scraping

IOT App = Node-RED + Snap4City examples
- Demo + exercises, Data processing with IOT Applications

Integration of External Services into IOT Applications
- Integration/Automating with: Ticketing Systems Workflow, Twitter Vigilance for social media analysis, CKAN open data portals
- Automated production of MicroServices for IOT app from External REST CALL APIs
- Integration with Telegram: SnapBot Solution

IOT Network Management and Control
- IOT Networks on Snap4City, Data Ingestion Strategy, IOT Broker Registration, IOT Brokers with Service paths, IOT Brokers with MultiTenant
- IOT Directory, IOT Device vs Time Series, IOT Device Model, IOT Device Management, IOT Device Registration, exploiting model

Complete examples
- Time Series, automating security, IOT device moving sensors
- Snap4City Self training sources

IOT Devices hardware-software integration
- Open and Proprietary devices, Open HW and Open SW, IOT Devices, IOT Gateway, IOT Edge (Arduino, Raspberry, etc.)
- (IBE CNR, Libelium, SIGFOX, Lora..), IOT Gateway/IoT Edge, IOT Tracking devices

IOT end-2-end Secure Stack, IOT ↔ Dashboards

Comparison with other Platforms and Fi-Ware
- FiWare & Snap4City

Acknowledgments

Snap4City (C), January 2022
Development Life Cycle
Smart City Services

Analysis & Design
- Analysis
- Design
- Data Discovery
- Data Ingestion

Data Analytics
- Data Analytics Development
- Special Tool Development
- IOT App Development
- Dashboard Development

Development
- Deploy
- Testing
- Publication Production
Levels of Difficulty

• Easy. non programmer level
• Moderate. Some JavaScript rudiment coding
• Good. JavaScript programming
• Golden. Programming in R Studio
• Professional. Exploiting Smart City API
• Excellent. Developing Full IOT Applications, Dashboard and Mobile Apps
IOT App for Smartening Solutions
URBAN PLATFORM: SMART CITY IOT AS A SERVICE AND ON PREMISE
Snap4City, Snap4Industry Architecture, V2

Data Sources, External Services
- PULL Data

Data Sources, Brokers, External Services
- Data Driven, Real Time

Data Ingestion, aggregation, regularization, reconcile:
- IOT Directory, NIFI, special tools

Knowledge base
- Semantic Reasoners

Indexing and aggregating
- OpenDistro (Elastic Search)

Data Analytics, Simulations, Special Tools
- R Studio, Tensor Flow, Python, …

IOT Applications, Business Logic
- Node-RED + Snap4City MicroServices

Federation
- Search and Query, Smart City API, Web Socket Server, GIS, Facet, semantic

Rendering, Acting, Acting, Widgets, Synoptics, MicroApps
- User interface, Drill down, maps, heatmaps

Inform, announce, Act!, warning, alarms, What-If, …

Authentication, Authorization, Platform & Processes Management, Data Inspector, Digital Twin, …
The usage of IOT Applications

- IOT Applications = Node-RED + Snap4City Libraries
- Used for:
  - Data Ingestion, Transformation, Extract, Load, and Adaptation (format and protocol), See Part 5 of the Course
  - IOT Edge Devices logic, for implementing logic on IOT Edge, including IOT Device control (see on Part 3 of the course, this part)
  - Business Logic control of Dashboards, via Web Sockets secure
    - see Part 2 of the Course
  - Control and schedule of Data Analytic, and Machine Learning (see part 4 of the Course)
  - Firing and condition identification and alerting.
Snap4City Architecture vs Data Ingestion

Data Ingestion

Data Sources, External Services
- PULL Data

Data Sources, Brokers, External Services
- Data Driven, Real Time

Data Ingestion, aggregation, regularization, reconcile:
- NiFi, IOT App

Knowledge base
- Semantic reasoners

Indexing
- OpenDistro (Elastic search)

Search and Query
- Smart City API
- Facet, semantic search

Data Analytics, Simulations
- R, Tensor Flow, Python

IOT Applications, Business Logic
- Node-RED + Snap4City MicroServices

Data Transformation and Business Logic

Front-End
- Rendering
- Acting
- Acting
- Widgets
- MicroApps
- User interface
- Interactive Dashboard
- Drill down
- heatmaps

Back-End
- Data Ingestion
- Knowledge base
- Semantic reasoners
- Indexing
- OpenDistro (Elastic search)
- Search and Query
- Smart City API
- Facet, semantic search
- Data Analytics, Simulations
- R, Tensor Flow, Python
- IOT Applications, Business Logic
- Node-RED + Snap4City MicroServices

Inform, announce, Act!, warning, alarms, What-IF, ..
Standards and Interoperability (2022)


https://www.snap4city.org/65
Snap4City IOT Brokers

Managing Public and Private IOT/IOE Devices

Snap4city Platform storage for «Data Shadow» and much more

Real Time

Real Time + Historical

Towards any IOT Device and/or Dashboard

Sensor Actuator

From Dashboard to IOT Device

From IOT Device to KB

Sensors

Actuator

Sensor Actuator

IOT Applications

Dashboards also provide rendering for sensor values

IOT Data Driven

IOT Applications

Snap4City (C), January 2022
IoT Devices

• Conceptually are IoT Devices with sensors/actuators, IN/IN-OUT
• They are classified in terms of nature/subnature
• For Searching and showing on maps and dashboards

HLT of IoT Devices can be:
- IoT Device Models, for example: «personal coffee machine»
- IoT Device name, for example: «mycoffemachine1», «CM23»
- IoT Device Variable, for example: «Temperature»

IoT Device Variables
- dateObserved: ..........
- ID:
- Status: ready
- Temperature: 70%
- WaterLevel: 35%
- UsedCapsBox: 30%
- Power: OK
- ....

Snap4City (C), January 2022
Mobile Devices

- They are a special case of IoT Devices
  - they are managed as IoT Devices in the system
- They are classified in terms of nature/subnature
- For Searching and showing on maps and dashboards, they are different

**HLT of Mobile Devices** can be:
- **Mobile Device Model**, for example: «sedan»
- **Mobile Device** name, for example: «BMW JD7356HD», «Ford KO786KK»
- **Mobile Device Variable**, for example: «velocity»

**Mobile Device Variables**
- **ID:**
- **dateObserved:** ............
- **Status:** ready
- **Temperature:** 70%
- **Gasoline:** 35%
- **Velocity:** 231.3 Km/h
- **Position:** 44.3223, 11.3432
- .....
POI, Point of Interest

- They are
  - classified in terms of nature/subnature
  - relevant services with codified metadata to simplify the massive management of huge amount of POIs
  - mapped on Knowledge Base on specific GPS location
  - Do not move over time
  - represented as PIN

- Do not have Time Series for variable over time
- May sporadically change over time
**HLT: Sensors-Actuators**

- Complex Event
- Dashboard-IOT App
- External Service
- Heatmap
- KPI (Key Performance Indicator)
- MicroApplication
- My Personal Data
- MyKPI
- MyPOI
- POI (Point of Interest)
- Sensor
- Sensor Actuator
- Special Widget
- Wfs (GIS)

**IOT Application**

**Dashboards**

- From IOT App to Dashboard
- From Dashboard to IOT App
Florence, Italy
Estimation of the mean waiting time at bus stops

Snap4City

BusLinesWaitTimeToDash

Firenze
Antwerp, Belgium
IOT App Smartening Devices and Dashboards
PaxCounter devices

- **Fix PaxCounter LoraWan**
  - sniffing on: Wi-Fi, Bluetooth
  - Sending data via LoraWan

- **Mobile PaxCounter LoraWan**
  - sniffing on: Wi-Fi, Bluetooth
  - Sending data via LoraWan

- **Fix PaxCounter, multiple out**
  - Sending data via LoraWan and Wi-Fi
  - sniffing on: Wi-Fi, Bluetooth

https://www.snap4city.org/456
Programmable PAX counting

Mobile PAXCounter 01 in Antwerp

Begin 3:00
Finish 5:30
Activate

Antwerp
IOT App Smart Parking
Smart Lonato del Garda

Smart Parking Monitoring

Lonato del Garda
IOT App Smart Industry 4.0
Snap4Industry
Industry Plant1.....

Admin

IoT Broker

IoT Devices/Edge

Fleet management

External Services

Internet

SECURE

IoT Broker

DCS

PLC

SCADA

PLC/RTU

IoT Devices/Edge

IoT Devices/Edge

IoT Broker

IoT Broker

Dashboards and Apps

Big Data Analytics, Artificial Intelligence

IOT Applications

Control and Supervision on Multiple Supply Chains

Industry 4.0 as a Service

https://www.snap4city.org/369
Snap4City Services also on IOT Edge!!!

IOT Networks

- IOT Gateways
- IOT Brokers

IOT Applications

- IOT Edge Devices
- IOT Brokers

Big Data Analytics, Artificial Intelligence

Dashboards and Apps

Mainly fog computing and NGSI V1, V2 with security
Prato
Smart City vs Industry 4.0

https://www.snap4city.org/758
GIDA set up

Smart City data from many sources

IOT Applications

Dashboards and Apps

IOT Data Shadow Snap4City

Big Data Analytics, Artificial Intelligence

ModBus to Snap4City Gateway Edge

5G network devices

Telemonitoring Telecontrol

https://www.snap4city.org/369
Controlling Energy Power

Measuring Energy Consumption

Alexa: Voice Commands

Measuring any kind of sensors values

Any kind of notification channel

Contextual (smart city/home) data, Data Analytics, Historical Data, Remote Control, Mobile App

IOT Edge: Node-RED + Snap4City

Local Control

DCS

Administrative Servers

ODBC

PLC

OPC UA

SCADA

Modbus

Any kind of notification channel

https://www.snap4city.org/369

Snap4City (C), January 2022
Snap4Altair Decision Support supervision and control, Industry 4.0

• Multiple Domain Data
  • Distributed Control System: energy, flows, storage, chemical data, settings, ..
  • Cost of energy, Orders,
  • Production Parameters
  • Maintenance data

• Multiple Levels & Decision Makers
  • Optimized planning on chemical model
  • Business Intelligence on Maintenance data

• Historical and Real Time data
  • Billions of Data

• Services Exploited on:
  • Multiple Levels, Mobile Apps, API

• Since 2020
Industry Plant Supervision and Maintenance

Aims

- **Control Room**: Higher level supervision and monitoring (since 2020)
  - Management of Production Plan Optimization
  - Control of Perimeter with drone and sensors
- **Maintenance ticketing** (since 2017)
  - *predictive* (in development)
  - 3D Digital Twin (in development)
MicroService Architecture

- IoT App/DA: Real Time & Stream Processing
- Predictive Maintenance
- Prod. Plan Optimization
- API/MicroServices
- Maintenance Intelligence
- Digital Twin Local / BIM
- Data Storage
- Management, Auth./Autoriz.

Energy Service

Data Connections and Transformation

Snap4City Dashboard Builder

DCS Real Time - Settimanale

Sinottico Sintesi Impianto Altair 2
Snap4City/Industry Detailed Architecture

- **IoT Edge**
  - Snap4City: Smart City API
  - MicroServices
- **Snap4City Dashboard Builder**
  - Data Storage
  - Orion Context Broker
  - GW-NGSI
  - IoT Devices
- **KeyCloak, LDAP**
- **Business Logic**
  - IoT APP
  - IoT APP
- **Data Analytic**
  - Predicting
  - Planning
- **DCS/SCADA**
  - Orders
- **Energy Services**
- **Transportation**
- **IoT Devices**
  - from the field
- **Administration**
- **External Services**
- **Production Parameters**
  - Dashboards, Visual Analytics, Synoptics, 3D, Maps
  - DCS Real Time, Settlorane
  - Sinottico Sintesi Implantato Attar 2
  - KeyCloak
  - LDAP
  - CKAN
  - GIS and ResM
  - BPM & BIM
  - IoT Directory
  - Snap4City Dashboard Builder

**Components**:
- **Snap4City**
- **IoT App**
- **IoT Devices**
- **IoT Devices from the field**
- **R Studio**
- **BPM & BIM**
- **GIS and ResM**
- **CKAN**
### Optimized Production Planner

#### Parameters

<table>
<thead>
<tr>
<th>Parameters (TabPar)</th>
<th>DCS (OCP-UA)</th>
<th>Administrative data (AS400)</th>
<th>Administrative Consolidated Planning data (AS400)</th>
<th>Energy data</th>
<th>Other Parameters</th>
<th>Planning result</th>
<th>Outcome</th>
<th>In production</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-09-25 18:47:36</td>
<td>2020-10-21 18:00:02</td>
<td>2020-10-21 17:59:47</td>
<td>2020-10-21 17:59:47</td>
<td>2020-10-21 23:00:00</td>
<td>2020-07-24 18:43:00</td>
<td>2020-10-21 18:00:12</td>
<td>completato</td>
<td>Sì</td>
</tr>
<tr>
<td>2020-09-25 18:47:36</td>
<td>2020-10-20 09:47:03</td>
<td>2020-10-20 09:47:45</td>
<td>2020-10-20 09:47:45</td>
<td>2020-10-20 23:00:00</td>
<td>2020-07-24 18:43:00</td>
<td>2020-10-20 09:47:21</td>
<td>completato</td>
<td>No</td>
</tr>
</tbody>
</table>
Produce Optimized Production Plan

Consolidated

List of Active Orders
Consolidated Orders
Last Params
Take last DCS
Take last Energy

Data Storage

Consolidated Orders

Optimized Production Planner

Possible Plans

Start Planning

Decision Support

Possible Plan

Production Plan
Some Flows
Västerås, Sweden
Smart Light Control of CAPELON

• Energy Domain
  • Smart Light, MQTT, ....
  • IoT Orion Broker FIWARE

• Dashboards
  • Map coverage on Sweden
  • Monitoring and real-time control
  • Energy control, analytics
  • Direct control

• Historical and Real Time data

• Services Exploited on:
  • Multiple Levels, API
  • Dashboards

• Since 2020
Dubrovnik, Croatia

https://www.snap4city.org/741
Dubrovnik

- Tourism Domain
  - Counting People
  - TV Cameras and WiFi
  - Social Media
- Dashboards
  - Monitoring and real-time control
  - People flow
  - Twitter Vigilance
- Historical and Real Time data
- Services Exploited on:
  - Dashboard
- Since 2020
Pont du Gard, France

https://www.snap4city.org/740
Pont du Gard

- Tourism Domain
  - KPIs
  - Social Media
  - People Flows
  - Bike Flows

- Dashboards
  - Monitoring KPI
  - People and bikes flows
  - Twitter Vigilance

- Historical and updated data

- Services Exploited on:
  - Dashboard

- Since 2020

Snap4City (C), January 2022
IOT App Smart Industry 4.0
ModBus Integration

Snap4City (C), January 2022
A large range of devices: sensors and actuators
• Over serial as RS485 and/or IP
IOT App vs Smart Home
Snap4Home
Prato
Smart City vs Smart Home Estra
https://www.snap4city.org/758
Sonoff: Controlling Energy Power
Philips Hue: Controlling Lights
Hue: Motion Control / Alarm
TP Link: Controlling / Measuring Energy Plugs
Alexa: Voice Control

IOT Edge: Raspberry pi: Node-RED + Snap4City

Measuring Temperature and Humidity
Garage Door
Window Roller Shutters
Controlling Motors
Alarm sound and light
Controlling Irrigators

Environmental Contextual data from the city
Historical Data, Remote Control, Mobile App

https://www.snap4city.org/620

Snap4City (C), January 2022
Example: IOT App on Snap4Home

Snap4City (C), January 2022
Snap4Home

Snap4Home 5G Demo

Hue Hub
Motion Control / Alarm
TP Link plugs: meter
Alexa: Voice Control

IOT Edge:
Raspberry pi:
Node-RED
Snap4City MicroService Library

Environmental Contextual data from the city.
Historical Data, Remote Control, Mobile App

5G gateway

Orion Broker
Dashboards
Advanced Smart City API

https://www.snap4city.org/369

Snap4City (C), January 2022
Snap4Home

IOT Edge:
Raspberry pi: Node-RED + Snap4City

Philips Hue: Controlling Lights
Hue Hub

Hue: Motion Control / Alarm

TP Link: Controlling / Measuring Energy Plugs

Alexa: Voice Control

Measuring: Temperature, Humidity, light in the room

Monitoring: CPU clock, status

5G gateway

Internet

Snap4Home 5G Demo

Environmental Contextual data from the city.
Historical Data, Remote Control, Mobile App

Snap4City (C), January 2022
Snap4Home 5G Demo

Thu 11 Jun 18:07:32

Gio 11 Giu
Prato
Pioggia e schiarite
18°C / 22°C
Powered by LaMMA
Ven 12 Giu
14°C / 27°C
Nuvole
Sab 19 Giu
13°C / 23°C
Nuvole
Dom 14 Giu
Temp N/A
Lun 15 Giu
Temp N/A
IOT App vs Smart City Solutions
Alert Registration

Alerting Generation

This dashboard contains data derived from actual sensors and predictive values under validation

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MzA0OQ==

Snap4City (C), January 2022
MicroServices Suite for Smart City

• Badii, C.; Bellini, P.; Difino, A.; Nesi, P.; Pantaleo, G.; Paolucci, M. MicroServices Suite for Smart City Applications.

• *Sensors* 2019, 19, 4798.

• [https://www.mdpi.com/1424-8220/19/21/4798/pdf](https://www.mdpi.com/1424-8220/19/21/4798/pdf)
Control Room Operator
Would like to:
- **Monitor** traffic flow,
  Environment, Car parking,
  Cycling, First aid, temp., ..
- **Act and** monitor Dynamic
  Plates
- **Act and** monitor red lights

Driver, Policeman
Would like to:
- Monitor traffic,
  Parking, traffic events,
  speed limit, ...
- **Act and** monitor red
  lights
Dashboards with city data and your data/actuators

Sensors:
- Values
- Status

Actuators:
- Buttons
- Dimers
- Etc.

Virtual Sensors and Actuators
IOT Application with City Dashboard
simple development
Reporting Critical Events

Control Room Operator
Would like to:
- **Monitor** events vs services in the city and receive critical event notifications from on the road operators.
- **Assess contextual condition**, services status

On the road operator
Would like to:
- Monitor data of traffic, Parking, environment, speed limit, services,
- **Send critical event notifications via coded description**
IOT Application with City Dashboard simple development
Other Cases

- Telecontrol for water depuration plant
- Traffic Flow computing in Florence
- Snap4Home: Casa Domotica
- Snap4Industry: industry control
- COVID 2019
- Computing of public transport quality
- Managing Heatmap production
- Managing Smart Parking
- Managing Smart Bike Sharing
- Telecontrol of Energy Plant
Andamenti Nazionali e Regionali infezione COVID-19

Sulla base dei dati della protezione civile, elaborazioni DISIT Lab

Per evidenziare gli andamenti di vostro interesse: eliminare le curve che non interessano selezionandole in legenda.

Alcuni dati in passato non sono pervenuti alla protezione civile

COVID 2019

Snap4City (C), January 2022
Moving IOT Devices / Sensors, Tracking Devices
Working with Sensor Data from Moving Devices

- Moving data can be collected by using:
  - **MyKPI**: in which each MyKPI has a `ValueName`, `Unit`, `Type`, etc.. And also GPS location
  - **IOT Device in Mobility**: which generates a new HLT SensorMobile which is partially developed so far
MyKPI: Tracking of Devices and Mobiles

- Real Time Trajectories for:
  - Mobile Phone
  - Moving IOT Devices
  - OBU, Vehicular Kits
  - Multiple tracks
  - Day by day

- Micro Application

---

TrackerFordOBD2

---

Apps

Mobile

PAX Counter

OBD2

OBU

IOT Device

MOBILE

Snap4City (C), January 2022
Tuscany in a Snap Mobile App on Android
PaxCounter devices

- **Fix PaxCounter LoraWan**
  - sniffing on: Wi-Fi, Bluetooth
  - Sending data via LoraWan

- **Mobile PaxCounter LoraWan**
  - sniffing on: Wi-Fi, Bluetooth
  - Sending data via LoraWan

- **Fix PaxCounter, multiple out**
  - Sending data via LoraWan and Wi-Fi
  - sniffing on: Wi-Fi, Bluetooth

https://www.snap4city.org/456
Programmable PAX counting

Mobile PAXCounter 01 in Antwerp

Begin 3:00
Finish 5:30

Antwerp

Snap4City (C), January 2022
Managing IOT Applications and Containers all
IOT Application Listing, they can be

- Basic (white)
- Advanced (red)
- IOT Edge
  - Raspberry Pi
  - Android
  - Win/Linux
- Data Analytic (Plumber)
- Web Scraper (Portia)
IOT Applications Listing

– Basic / Advanced
– On IOT Edge Raspberry Pi
– On IOT Edge Android
– On IOT Edge Win/Linux
IOT Application Self Control

• Properties
  – Name, Type, Creation date

• Control
  – Restart Container
  – Delete IOT App

• Change of ownership
  – Pass to another Snap4City User

• From inside the IOT App
  – Restart
  – Update Snap4City Library

Automating restart and update
Remote Control of IOT Applications on IOT Edge Devices
Snap4City Services also on IOT Edge!!!
Why it is useful

• **Need to transform** data from your local devices to Snap4City devices or data storage or dashboards,

• **Need to play the role of multiprotocol HUB** as in:
  – **Snap4Home**: *Scenario: Snap4Home, how to exploit Snap4City solution on home automation*
  – **Snap4Industry**:
    • *Scenario: High Level Control of Industrial Plant*
    • **Snap4Industry: Snap4City for Industry 4.0**
    • *Scenario: 5G Enabled Water Cleaning Control*
IOT Edge Device for Snap4City

• Computer based solutions with *Node-RED + Snap4City Library*

• Node-red supports:
  – Raspberry pi, Linux based, Windows based, android with Termux, and also on a several servers. [https://nodered.org/docs/getting-started/](https://nodered.org/docs/getting-started/)

• Snap4City Library:
  – From Node-RED: “manage palette” in the main menu’.
  – From Node-RED library: [https://flows.nodered.org/search?term=snap4city](https://flows.nodered.org/search?term=snap4city)

• *You can add any kind of protocol and interface to IOT Edge Device*
Remote Control/Management of IOT Edge

IOT Edge Device

- On premise, on your local plant/home
- Local Access on IOT Edge Device on your local network
- On Cloud
  - Internet
  - FireWall Gateway
  - Intranet

Click on it

Remote Access/Control/program to your IOT Applications via secure connection

Snap4City (C), January 2022
Advantages of IOT Edge remote control/program

• You do not need to be/move in the local network to access at Your IOT Edge Devices for programming or maintenance, SINCE With Snap4City:
  – You can update the logic flow of your IOT Edge Devices from remote,
  – You can perform remote maintenance of your IOT Edge Devices and programms without moving from your office

AND

• You can access to the IOT edge from remote by using a secure connection
• You can activate the remote control feature singularly for each IOT Edge Device
• You do not need to reconfigure your Firewall or Gateway in most of the cases
IOT Edge Device

IOT Devices

Other Local Connections

IOT Brokers

HTTPS

IOT Brokers

HTTPS

HTTPS

WSs

Remote Access/Control/program to your IOT Applications via secure connection

IOT Edge:
Node-RED + Snap4City

toward internet

Firewall Gateway

Search and Query, Smart City API
Face, semantic search

Knowledge base
Semantic reasoners

Indexing and aggregating
Elastic search

Snap4City (C), January 2022
HOW To install IOT Edge Remote Control feature

• The installation is very simple
  1. install Snap4City basic library
  2. Drag and drop block from S4CUtility
  3. Configure the block with your credentials
  4. Deploy of the IOT App
  5. Go in the list of Your IOT Applications on Snap4City.org or other cloud or on premise installations
  6. Identify the IOT Edge IOT App and click on it to open the view on the IOT Applications flows
Creating IOT Applications with Node-RED
IOT Applications Development

MicroServices collections

My IOT Applications

IOT App. Editor

Generating IOT App With Dashboard

Sharing/saving reusing IOT App

Resource Manager

ServiceMap Discovery

Dashboard Collection, Editor and Wizard

Knowledge Base, Km4City
# Basic Node.js Blocks on NodeRed on our Advanced IOT Apps

<table>
<thead>
<tr>
<th>common</th>
<th>network</th>
<th>sequence</th>
<th>social</th>
<th>dashboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>inject</td>
<td>mqtt in</td>
<td>split</td>
<td>email</td>
<td>button</td>
</tr>
<tr>
<td>debug</td>
<td>mqtt out</td>
<td>join</td>
<td>twitter</td>
<td>dropdown</td>
</tr>
<tr>
<td>!</td>
<td>http in</td>
<td>sort</td>
<td>email</td>
<td>switch</td>
</tr>
<tr>
<td>complete</td>
<td>http request</td>
<td>batch</td>
<td>twitter</td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>http in</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>catch</td>
<td>http request</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>websocket in</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>link in</td>
<td>websocket out</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>link out</td>
<td>comment</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Function
- function
- switch
- change
- range
- template
- delay
- trigger
- exec
- md5
- soap request
- request
- string
- xml converter
- random
- rbe

## Input
- amqp in
- amqp2 in
- stomq in
- delay
- trigger
- exec
- md5
- soap request
- request
- string
- xml converter
- random
- rbe

## Output
- amqp out
- amqp2 out
- stomq out
- delay
- trigger
- exec
- md5
- soap request
- request
- string
- xml converter
- random
- rbe

## Storage
- file
- worldmap
- turf
- worldmap in
- audio out
- notification
- ui control
- mongo file
- mysql
- tail

## Network
- ping

---

**Snap4City (C), January 2022**

**+ on IOT Edge Raspberry**
Node-RED Basic Blocks

It is provided with a minimum set of functionalities (the building blocks/nodes) while other blocks can be easily added loading them from a large library made available by the JS Foundation.

Despite its diffusion, for the usage in the context of Smart City it was not powerful to cope with the basic requirements of the domain.

The classical nodes provided in the standard version can be classified as: input, output, function, social, storage, analysis, advanced, and dashboard.
Divides the input message into multiple messages as indicated in the configuration. If you have an array at the input, you can configure it to send each element of the array individually at the output.

Treads the input message on possible different outputs based on a comparison made on the input message.

Operates in reverse order to the split. Joins the incoming messages in the mode indicated in the configuration.
Hello World of Node-RED

Node-RED

- Node-RED is a flow-based development tool for visual programming proposed by JS Foundation.

- The Node-RED approach is a mix of visual composition of nodes/blocks to compose the so-called flows that are concurrently executed by an engine Node.js.

- It is quite diffuse being also directly provided into official releases of IOT devices as Raspberry Pi family.

- Based on Node.js

- 100% open source

Snap4City (C), January 2022
DEMO
Section 1
Node-RED Hello World
Example of simple IOT Application

In this demo let's create an IoT Application that:

• generate a random value,
• the value is switched on the correct path
• the value is showed in the local dashboard of NodeRed
Nodes for flow

Generates an input for the other nodes. It can be repeated at predefined intervals, entered manually and of various types (timestamp, string, number, boolean, JSON, etc).

Each message that enters the debug node is shown in the "debug" tab on the right of nodered (you can choose which part of the message to show).

Generates a random number. You can configure the number generation interval and the type (integer or float).

Evaluates the input message and routes it to the correct output according to the desired configuration.

Shows a number inside a gauge counter.

Shows a text inside the local dashboard.
Step 1

- Inject and Debug
- Connect
- Configure
- Deploy
- Click
- Observe
Step 2

- Random
- Connect
- Configure
- Deploy
- Click
- Observe

Snap4City (C), January 2022
Step 3

- Switch
- Connect
- Configure
- Deploy
- Click
- Observe
Step 4

- Gauge and text
- Connect
- Configure gauge
Step 4 Bis

• Gauge and text
• Connect
• Configure text
Step 5

- Deploy
- Click
- Click
- Observe
Nodes configuration 2/2
Nodes connections
Explaining: IOT Application Flow

• On Click or Every 15 minutes the *timestamp* node sends a message to the *random* node.

• When the message arrives, the *random* node generates a random number as output message.

• The *switch* node routes the value on the correct output based on the configuration.

• The Number can be sent to Different kinds of nodes to show it on NodeRed Dashboard.
This is a local Node-RED dashboard.

The dashboards created within the Snap4city platform are more:

- Powerful
- Flexible
- Secure
end DEMO

Section 1
SCALABLE SMART ANALYTIC APPLICATION BUILDER FOR SENTIENT CITIES

IOT App = Node-RED + Snap4City
In the IOT Application of Snap4City, it is possible to:

- Execute flows that process data as: Event Driven, Batch (periodic or not)
- Create multiple concurrent Flows for each IOT Application
- Create subflows as macros to be reused
- Create Groups of nodes as macro
- **Save/load, share**, of nodes, flows and applications with other users via
  - the Snap4City Resource Manager or
  - with JS Foundation or
  - via email, skype, file sharing in general
  - ..
In the IOT Apps of Snap4City, it is possible to Extend the Capabilities:

- Load other Nodes, segments of flow and entire flows from several sources: email, libraries, S4C repository, etc.
- Load other libraries of MicroServices/Nodes/Blocks from Manage Palette
  - A large set of Libraries of Node is available.
  - The loading may have some limitations for security reasons
- Get more IOT Apps above the Limit that may depend on the organization and/or on personal authorizations, ask to Admin
Load Library from Palette

Two views of the same libraries

https://flows.nodered.org/
Load an IOT application of example Snap4City (C), January 2022
MicroServices Areas

- IOT Sectors and Actuators Services
- Special Services for IOT Edge, IOT Button

Big Data Storage Services

Knowledge Services: maps, search, discovery

Access Services

Application Development and Management

Computation Management Services

Big Data Analytic Services: Machine learning, R Studio, Python

IOT Applications ETL Process Services, automating data ingestion and transformation

External Services

Micro Services

Workflow Ticketing

Notification Services

Dashboard Services

App Services

MicroApplications Development Kit

https://flows.nodered.org/?term=snap4city

Daisy of MicroServices

Snap4City (C), January 2022
Smart City Entities Search: search and access to city entities and their relationships in the city.

Historical Data: search and access to data collected over time into the smart city data aggregator.

Save and Get Personal Data: for many smart city applications, the possibility of saving and retrieval of personal data enables a large variety of smart scenarios for the final users and operators.

Advanced Dashboards: This means to have the possibility of developing a real user interface of the IOT App (to render and produce data for the IOT network).

Data Analytic: The real need in the context of smart City is to have the possibility for a data-analysts of creating some data analytic processes and use it into the flow as MicroService without the intervention of a programmer nor administrator.

IOT Device Connection: This means that the developers expect to have the possibility of using nodes for connecting to a large set of IOT devices using different protocols, and thus connecting to different kind of IOT brokers.

IOT Directory: It should be a single point service for searching, managing and discovering all the IOT Devices which can be connected to the infrastructure by means of a large set of heterogenous IOT Brokers.
IOT Applications

IOT Applications = Node-RED + Snap4City Platform

• A collection of more than 150 MicroServices have been developed covering the above-mentioned requirements and much more.
• The issue was not only to formalize the MicroServices, but also to create the infrastructure that enable their usage. In many cases, the simple MicroServices hide very complex and sophisticate tools and algorithms (Snap4city Platform).
• They are formally distributed as two official libraries of Node-RED nodes (Snap4City Basic and Advanced) by the JS Foundation portal.
• They can be directly installed in any Node-RED tool of any operating system.
Basic Node.js Blocks on NodeRed on our Advanced IOT Apps

+ on IOT Edge Raspberry

Snap4City (C), January 2022
**IoT Applications**

- **Data ingestion**: more than 70 protocols IOT and Industry 4.0, web Scraping, external services, any protocol database, etc.
- **Data access**: save/retrieve data, query search on expert system, georeverse solution, search on expert system Km4City ontology, etc.
- **Data Transformation/transcoding**: binary, hexadecimal, XML, JSON, String, any format
- **Integration**: CKAN, Web Scraping, FTP, Copernicus satellite, Twitter Vigilance, Workflow OpenMaint, Digital Twin BIMServer, any external service REST Call, etc.
- **Manipulation of complex data**: heatmaps, scenarios, typical time trend, multi series, calendar, maps, etc.
- **Access to Smart City Entities and exploitation of Smart City Services**: transport, parking, POI, KPI, personal data, scenarios, etc.
- **Data Analytic**: managing Python native, calling and scheduling Python/Rstudio containers as snap4city microservices (predictions, anomaly detection, statistics, etc.)
- **User interaction on Dashboard**: get data and message from the user interface, providing messages to the user (form, buttons, switches, animations, selector, maps, etc.)
- **Custom Widgets**: SVG, synoptics, animations, dynamic pins on maps, etc.
- **Event management**: Telegram, Twitter, Facebook, SMS, WhatsApp, CAP, etc.
- **Hardware Specific Devices**: Raspberry Pi, Android, Philips, video wall management, etc.
Two Snap4City Libraries

https://flows.nodered.org/search?term=snap4city
Two Snap4City Libraries

We suggest also to install:

AND: From Resource Manager

https://flows.nodered.org/search?term=snap4city
MicroServices Suite for Smart City Applications

- https://doi.org/10.3390/s19214798
- https://www.mdpi.com/1424-8220/19/21/4798/pdf
IOT App = Node-RED + Snap4City

search vs services, the ServiceURI
- Street and geoinformation of the territory and details for routing, navigation, ...
- GeoResolution, Environmental data
- Mobility and Transport: public and private, public transport, parking status, fuel stations prices, traffic sensors, etc.
- Culture and Tourism: POI, churches, museum, schools, university, theatres, events in Florence
- Environmental: pollution real time, weather forecast, etc.
  - Environmental data geo resolution
- Social Media: twitter data
- Health: hospital, pharmacies, status of the first aid triage in major hospitals, ...
- Alarms: civil protection alerts, hot areas, ...
Access to Point of Interest information, POI

- **POI**: point of interest
- **type**: macro (nature) and subcategories (subnature)
- **Position**: GPS, address, telephone, fax, email, URL, ...
- **Description**: textual, multilingual, with images, ...
- **Link** to dbPedia, Linked Open Data
- **Links to other services**
- **Real time data if any**: sensors data, timeline, events, prices, opening time, rules of access, status of services, status of queue, etc..

- *See transversal services on ServiceMap*
  - Regular and in test platform
A SKOS area into the Km4CIty Ontology and Knowledge base for modeling POI and any element on map.
• **ANY kind of sensors**

• **To Get DATA of a Service / POI /sensor**
  – Historical and real time
  – Real Time

Snap4City (C), January 2022
• **Distance from GPS point**

• **Point is in Polygon?**
  – Polyline as WKT
For example to search for:

- **POIs:**
  - near a GPS position, from text, along a path, in an area, etc..
  - Public Transport information / data
  - Suggestions
  - Public Transport Means Routes/Paths
  - Events in the area
  - Value Type (kind of data)
  - Etc.

- **To Get DATA of a Service / POI /sensor**
  - Real Time
  - ANY kind of sensor
IOT Discovery on IOT Application Development

Snap4City (C), January 2022
Data Registration Flow at a Glance

IOT Directory:
Devices...
Sensors..
Actuators...

Knowledge Base,
ServiceMap,
SmartCity API,
ASCAPI

DataInspector
Dashboard Wizard

IOT Apps
## Notation Terminology

<table>
<thead>
<tr>
<th>WHERE</th>
<th>Are synonymous at level of service which can be <strong>IOT device or entity</strong> with data</th>
<th>Are synonymous at level of the single attribute of the entity, device, service, etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOT Directory</td>
<td>IOT Device</td>
<td>Sensor, Actuator, Attributes, Values (value name)</td>
</tr>
<tr>
<td>Knowledge Base, ServiceMap, SmartCity API, ASCAPI</td>
<td>Service, ServiceURI, SURI</td>
<td>Attribute, Metric</td>
</tr>
<tr>
<td>Datalnspector, Wizard, Dashboard</td>
<td>ValueName</td>
<td>Sensor, Sensor Actuator, ValueType</td>
</tr>
<tr>
<td>IOT Applications, Node-RED</td>
<td>ServiceURI, SURI</td>
<td>SURI and its real time results of the objects into the data structure</td>
</tr>
</tbody>
</table>

**ServiceURI, SURI of a sensor device:**
- [http://www.disit.org/km4city/resource/METRO759](http://www.disit.org/km4city/resource/METRO759)
- [http://www.disit.org/km4city/resource/iot/orionCAPELON-UNIFI/CAPELON/Streetlight%3A90FD9FFFEBD5A7F](http://www.disit.org/km4city/resource/iot/orionCAPELON-UNIFI/CAPELON/Streetlight%3A90FD9FFFEBD5A7F)

**ServiceURI, SURI extended with attribute/variable/value:**
- [http://www.disit.org/km4city/resource/METRO759&metric=vehicleFlow](http://www.disit.org/km4city/resource/METRO759&metric=vehicleFlow)
- In some cases
  - [http://www.disit.org/km4city/resource/METRO759/vehicleFlow](http://www.disit.org/km4city/resource/METRO759/vehicleFlow)
IOT Directory

• For: IOT Devices, Sensors, Sensor mobile, Actuators, Virtual Sensors, etc.

• Accessible as
  – ServiceURI
  – Device URI
• For PUBLIC:
  – IOT Devices, Sensors,
  – Sensor mobile,
  – Actuators,
  – Virtual Sensors,
  – POI, etc.
• See as
  – ServiceURI

Snap4City (C), January 2022
# Understanding / Testing an IOT Device

<table>
<thead>
<tr>
<th>AdminDevice001</th>
<th>orionUNIFI</th>
<th>Ambiental</th>
<th>MYOWNPRIVATE</th>
<th>active</th>
<th>EDIT</th>
<th>DELETE</th>
<th>VIEW</th>
</tr>
</thead>
</table>

**Broker URI:** https://broker1.snap4city.org  
**Kind:** sensor  
**Device Type:** Ambiental  
**Protocol:** ngsi  
**Model:**  
**Longitude:** 9.228193  
**Device URI:** http://www.disit.org/km4city/resource/iot/orionUNIFI/AdminDevice001  
**Organization:** DISIT  
**Payload NGSI V1:**  
| b7c-f25c-4cb6-95eb-e4b363222bef  
| Created: 2018-05-24 21:54:03  

**Broker Port:** 8080  
**Visibility:** MyOwnPrivate  
**Format:** json  
**Producer:** Raspberry Pi  
**Latitude:** 45.499369  

See Payload NGSI V1 in JSON directly from the Broker, Last message of the broker.  
See Payload NGSI V2 in JSON directly from the Broker, Last message of the broker.  
See IoT Device on ServiceMap.  
Create a Message to be sent at the IoT broker regarding this device.
Knowledge Base view

Data Inspector

Wizard

Some functionalities are limited to certain roles

Snap4City (C), January 2022
• For example to search for:
  – POIs:
    • near a GPS position, from text, along a path, in an area, etc..
    – Public Transport information / data
    – Suggestions
    – Public Transport Means Routes/Paths
    – Events in the area
    – Value Type (kind of data)
    – Etc.

• To Get DATA of a Service / POI /sensor
  – Real Time
  – ANY kind of sensors

• Distance from GPS point
• Similar to basic Search functions but with more flexibility of the function for programming the search
• Adding Dynamic behavior:
  – Getting in input JSON with parameters
• To Get DATA of a Service / POI /sensor
  – Historical and real time
  – ANY kind of sensors
IOT App = Node-RED + Snap4City

IOT Devices NGSI just list see later
- **Search** for IOT Devices in a given area, or for kind (temperature, model, location, producer, Broker, ...)
- **Subscribe** to one or more IOT Devices independently on their protocol, broker, owner, etc.
- **Send** data to IOT devices
- Establish with IOT Devices **Secure** certified Connections

- Please note that many other protocols can be also added, adding mode nodes, or registering IOT brokers to the Snap4City IOT Directory
IOT App = Node-RED + Snap4City

Others nodes
- Save and retrieve MyKPI into the safe personal data storage
- Access to MyKPI and to those that other user have delegated to Me
- **MyKPI are:**
  - Time series of data with GPS coordinates that can change over time
  - Suitable for: moving sensors, trajectories, data from OBU, data from mobile, sensor data (if needed), etc. etc.
- **MyPOI are:**
  - POI with full metadata description and static coordinates
• Request metrics from Twitter Vigilance Channel service and engine of DISIT Lab
  – Different Twitter Vigilance services may be attached according to the Organization, different metrics and values

• Location services
• Maps and get position (raw solution)

• Getting data from DataGate/CKAN
• Publishing data to DataGate/CKAN
• Managing time series on DataGate/CKAN
Third party solution to: Control Maps from IOT Apps

- Show points on maps
- Get Points
- Tracks
- See examples on:
  - https://iot-app.snap4city.org/nodered/nrve0e3/ui/#!/0
  - https://www.snap4city.org/409
  - https://www.snap4city.org/417

Snap4City (C), January 2022
IOT App = Node-RED + Snap4City
Dashboard Integration
Native Local

• Input/output
• non secure
• Limited in graphics
• No authentication
• No HLT
• No integration
• No historical data
• No Synoptics
• Etc..

• Local on IOT Edge

Snap4City

• Input/output
• Secure
• Advanced in graphics
• Single Sign On
• Several HLT
• Fully integrated
• Historical data
• Full Synoptics
• Etc..

• Remote for IOT Edge via WebSocket Secure
From Dashboard to IOT Devices/App

- **Widgets:**
  - Impulse Button
  - Button
  - Switch
  - Dimer/Knowb
  - KeyPad
  - Geolocator
  - Selection/Dropdown
  - Form
  - Map Picking

- **Registered** on some IOT brokers with NGSI mutual authentication
Dashboard-iot App

From Dashboard to IOT App

MyKPI variable onchange
MapClick
Synoptics

IOT Application

Impulse button
Numeric keyboard
Switch button
dimmer
Geolocator
dropdown
Form
Coordinates from map
Event driven
My kpi
Synoptic read
Synoptic subscribe
Geolocation of Mobile Device

- Complete message
  - Returns a JSON containing all information about geolocation
- Latitude
  - Returns the latitude
- Longitude
  - Returns the longitude
- Accuracy
  - Returns the accuracy of latitude and longitude
- Altitude
  - Returns the altitude
- Altitude Accuracy
  - Returns the altitude accuracy
- Heading
  - Returns the heading
- Speed
  - Returns the speed

Web Browser GPS data rendering the Snap4City Dashboard can be passed to IOT Applications and saved 😊
Multi Data Map GPS Location Picking vs IOT App

1) Click
2) GET event with:
   -- Lat,Long
   -- ServiceURI

• 3) The click on the map passes GPS coordinates into IOT App. Thus you can use them to:
   – search for location
   – picking the value of one or more heatmaps
   – dynamically change data on widgets and dashboards
   – Etc.

Snap4City (C), January 2022
Multi Data Map ServiceURI selection vs IOT App

1) Click on PIN

2) GET event with:
   - Lat,Long
   - ServiceURI

3) The click on the map passes GPS coordinates into IOT App and the ServiceURI. Thus you can use them to:
   - search for location
   - picking the value of one or more heatmaps
   - dynamically change data on widgets and dashboards
   - Get all the ServiceURI information and exploit them on Business Logic
   - Etc.
SVG Custom Widgets Examples 2

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=Mjk4Ng==
msg.payload = {value:JSON.parse(msg.payload).selected};
return msg;
msg.payload = { "form": { "options": [
    { "label": "enter text", "value": "", "type": "text", "required": true },
    { "label": "enter number", "value": "", "type": "number", "required": false },
    { "label": "enter email", "value": "", "type": "email", "required": false },
    { "label": "enter password", "value": "", "type": "password", "required": false },
    { "label": "enter check", "value": "checked", "type": "checkbox", "required": false },
    { "label": "enter check2", "value": "", "type": "checkbox", "required": false },
    { "label": "enter switch", "value": "on", "type": "switch", "required": false },
    { "label": "enter switch2", "value": "", "type": "switch", "required": false },
    { "label": "enter date", "value": "", "type": "date", "required": false },
    { "label": "enter time", "value": "", "type": "time", "required": true }
    ], "selected": [] } }
return msg;
msg.payload = { "form": { "options": [
{ "label": "enter text", "value": "", "type": "text", "required": true },
{ "label": "enter number", "value": "", "type": "number", "required": false },
{ "label": "enter email", "value": "", "type": "email", "required": false },
{ "label": "enter password", "value": "", "type": "password", "required": false },
{ "label": "enter check", "value": "checked", "type": "checkbox", "required": false },
{ "label": "enter check2", "value": "", "type": "checkbox", "required": false },
{ "label": "enter switch", "value": "on", "type": "switch", "required": false },
{ "label": "enter switch2", "value": "", "type": "switch", "required": false },
{ "label": "enter date", "value": "", "type": "date", "required": false },
{ "label": "enter time", "value": "", "type": "time", "required": true }
], "selected": [] }
}
return msg;

"selected":["a text","123","paolo.nesi@unifi.it","aaaaaa","checked","","on","","2021-03-19","09:38"]
Dashboard-IOT App

From IOT App to Dashboard

IOT Application

Nature

Snap4City (C), January 2022
Single Content Widget (flexibility)

From Dashboard Editor and IOT Applications, accepts in input:

• Numbers
• String
• HTML code

https://www.snap4city.org/578
Send Voice Messages on Dashboards

• Connectable on all «String» Variables
• Controllable from IoT Applications
• Simple Play button on Dashboard Widget
• Configured as:
  – Voice Language
  – Voice timbre: male, female, ...
  – Voice Tone
  – Voice Volume
  – ...

Snap4City (C), January 2022
Dynamic Widgets data on Dashboard from IOT Applications

Snap4City (C), January 2022
How the Dashboards exchange data

- **ServiceMap**
  - Super ServiceMap
  - Req. ServiceURI

- **Snap4City BigData Storage and KB**
  - API, External Services, MicroApp
  - MyKPI, MyPOI, ...
  - Metric, KPI
  - IOT Broker Orion Quantum Leap

- **API, External Services, MicroApp**
  - Req. MyKPI ID
  - Traffic Flow, MAPS, Heatmaps
  - GIS, HTTPS URLs

- **IOT Application**
  - Req. KPI, Metric ID
  - Dynamic Data, computed into IOT Application
  - Rx. Dynamic Data
  - Event Driven Synoptics
  - Actions, Show

- **Dashboards**
  - Snap4City (C), January 2022
<table>
<thead>
<tr>
<th>Widgets ICONS</th>
<th>Widget Name, Description</th>
<th>IOT App</th>
<th>Dashboard-IOT App</th>
<th>KPI (metric)</th>
<th>MyPersonalData</th>
<th>MyData</th>
<th>MyKPI</th>
<th>Sensor</th>
</tr>
</thead>
<tbody>
<tr>
<td>XX</td>
<td>Single Content</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Speed Limit (see custom for more)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Speedometer</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Gauge</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Single Bar, V/H</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Single and Multiple Bars, stacked or not, ordered</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MultiSeries, shaded, stacked and non stacked, TTT</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Time Trend (single)</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Time Trend Compare</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SpiderNet, radar, Kiviat</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pie, Donut, 2 layers Donut</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Calendar</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• IOT APP column in previous table:
  – X: means that from the IOT App you can send a new value or array to the widget directly, without the need to have is stored into Sensor or MYKPI variable, etc.
  – CS, widget supports Change (data) Source, in the sense that: from the IOT App is possible to send a command to the Widget to change the data source. E.g., selecting sources among: Sensors (serviceURI), MyKPI (ID), any value produced on the IOT App directly. (cs) recent additions

• Dashboard IOT App column in previous table:
  – X: there is a MicroService / node on IOT App to act on those widgets on dashboard. The data are visualized.
  – DD, widget is Data Driven, in the sense that new data in push can be sent and the widget is updated in real time on web page without web page reloading

**TC4.9: New Support Widgets for Bars, Barseries, Trend, and Series, on Dashboards and IOT Applications** (partially obsolete)
Dynamic Widget data

- **ServiceURI (ID)**
  - metricId: "http://www.disit.org/km4city/resource/tusc_weather_sensor_pl_3366348"
  - metricHighLevelType: "sensor"
  - metricName: "tusc_weather_sensor_pl_3366348"
  - metricType: "airTemperature"

- **ServiceURI (ID)**
  - metricId: "https://servicemp.disit.org/webopeneo/api/v0/services?serviceURI=http://www.disit.org/km4city/resource/tusc_weather_sensor_pl_3385232&format=json"
  - metricHighLevelType: "sensor"
  - metricName: "tusc_weather_sensor_pl_3385232"
  - metricType: "airTemperature"

- **MyKPI (ID)**
  - metricId: "17056230"
  - metricHighLevelType: "MyKPI"
  - metricName: "S1DMtuscanyTrackerLocation"
  - metricType: "Velocity"

- **Dynamic**
  - metricId: ""
  - metricHighLevelType: "Dynamic"
  - metricName: "BatteryTemperatureGalaxyNote"
  - metricType: "Gradi Centigradi"
  - metricValueUnit: "°C"
  - measuredTime: "2019-11-22T14:53:00Z"
  - value: 30.88998113104505

- **Dynamic**
  - metricId: ""
  - metricHighLevelType: "Dynamic"
  - metricName: "BatteryTemperaturearmi"
  - metricType: "Gradi Centigradi"
  - metricValueUnit: "°C"
  - measuredTime: "2019-11-22T14:53:00Z"
  - value: 32.8907/18741156

100% of Dynamic VECTs

**TC4.9: New Support Widgets for Bars, Barseries, Trend, and Series, on Dashboards and IOT Applications**

Snap4City (C), January 2022
Example of: Dynamic Widgets data on Dashboard from IOT Applications
How to send the Dynamic Data to Widgets

• TC4.9: New Support Widgets for Bars, Barseries, Trend, and Series, on Dashboards and IOT Applications

• Dynamic Data are used to control the Widget from the IOT App. To dynamically change:
  • ServiceURI (as metricID) to change the data source of a Dashboard Widget
  • MyKPI (as metricID) to change the data source of a Dashboard Widget
  • Dynamic, data computed somehow into IOT App, and sent to some Dashboard Widget without to save them on some Storage
Dashboard with Dynamic Data Managed by IOT App

Dynamic Widget data

The IOT App controlling the Dashboard data
msg.payload = [
    {        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3166540",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3166540",
        "metricType": "airTemperature"
    },
    {        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3182522",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3182522",
        "metricType": "airTemperature"
    },
    {        "metricId": "17057447",
        "metricHighLevelType": "MyKPI",
        "metricName": "OBD2 - Intake Air Temperature",
        "metricType": "Temperature"
    },
    {        "metricId": "17056579",
        "metricHighLevelType": "MyKPI",
        "metricName": "OBD2 - Intake Air Temperature",
        "metricType": "Temperature"
    },
    {        "metricId": "",
        "metricHighLevelType": "Dynamic",
        "metricName": "BatteryGalaxyNote",
        "metricType": "Temperature",
        "metricValueUnit": "°C",
        "measuredTime": "2019-11-21T14:51:00Z",
        "value": 42
    },
    {        "metricId": "",
        "metricHighLevelType": "Dynamic",
        "metricName": "Storage",
        "metricType": "Space",
        "metricValueUnit": "Gb",
        "measuredTime": "2019-11-21T14:51:00Z",
        "value": 12
    }
]
return msg;

Sensors, MyKPI and dynamic in this case

May be aggregated by Metric Type or by ValueName, staked, oriented .. see More Options

Snap4City (C), January 2022
msg.payload = [
    { "mapName":"15MinIndex_FastMobilityIndex", "metricName":"15Min Indexes", "value":1,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Fast Mobility" },
    { "mapName":"15MinIndex_GovernmentServicesIndex", "metricName":"15Min Indexes", "value":1,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Government Services" },
    { "mapName":"15MinIndex_HealthIndex", "metricName":"15Min Indexes", "value":1,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Health" },
    { "mapName":"15MinIndex_AverageIndex", "metricName":"15Min Indexes", "value":3,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Average" },
    { "mapName":"15MinIndex_HousingIndex", "metricName":"15Min Indexes", "value":5,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Housing" },
    { "mapName":"15Min Indexes", "metricName":"Max Value", "value":5,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Fast Mobility" },
    { "mapName":"15Min Indexes", "metricName":"Max Value", "value":5,
        "metricHighLevelType":"Dynamic", "metricValueUnit":"", "metricType":"Fast Mobility" }
]
return msg;

All dynamic in this case
msg.payload = [
    {
        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3166540",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3166540",
        "metricType": "airTemperature"
    },
    {
        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3176959",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3176959",
        "metricType": "airHumidity"
    },
    {
        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3166540",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3166540",
        "metricType": "airHumidity"
    },
    {
        "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3176959",
        "metricHighLevelType": "Sensor",
        "metricName": "tusc_weather_sensor_ow_3176959",
        "metricType": "airTemperature"
    },
    {
        "metricId": "17056579",
        "metricHighLevelType": "MyKPI",
        "metricName": "OBD2 - Intake Air Temperature",
        "metricType": "Temperature"
    },
    {
        "metricId": "",
        "metricHighLevelType": "Dynamic",
        "metricName": "My Home",
        "metricType": "airTemperature",
        "metricValueUnit": "°C",
        "measuredTime": "2019-11-21T14:51:00Z",
        "value": 31
    },
    {
        "metricId": "",
        "metricHighLevelType": "Dynamic",
        "metricName": "My Home",
        "metricType": "airTemperature",
        "metricValueUnit": "°C",
        "measuredTime": "2019-11-21T14:51:00Z",
        "value": 28
    },
    {
        "metricId": "",
        "metricHighLevelType": "Dynamic",
        "metricName": "My Home",
        "metricType": "airHumidity",
        "metricValueUnit": "%",
        "measuredTime": "2019-11-21T14:51:00Z",
        "value": 25
    }]

return msg;

Sensors, MyKPI and dynamic in this case
Multi TimeSeries

```javascript
var now = new Date();
var base = 60*60*100;
msg.payload = [
   {
      "serviceUri": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3166540",
      "metricHighLevelType": "Sensor",
      "metricName": "tusc_weather_sensor_ow_3166540",
      "smField": "airTemperature" // as MetricType
   },
   {
      "serviceUri": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3176959",
      "metricHighLevelType": "Sensor",
      "metricName": "tusc_weather_sensor_ow_3176959",
      "smField": "airTemperature"
   },
   {
      "serviceUri": "17057458",
      "metricHighLevelType": "MyKPI",
      "metricName": "Room 1",
      "metricType": "Temperature"
   },
   {
      "metricId": "",
      "metricHighLevelType": "Dynamic",
      "metricName": "BatteryTemperatureGalaxyNote",
      "smField": "Gradi Centigradi",
      "metricValueUnit": "°C",
      "values": [
         [now-64*base, 19.5], [now-60*base, 20.0], [now-56*base, 20.5], [now-52*base, 18.5], [now-48*base, 19], [now-44*base, 18.5], [now-40*base, 21.5], [now-36*base, 22.0], [now-32*base, 19], [now-28*base, 17.5], [now-24*base, 16.5], [now-20*base, 17.0], [now-16*base, 18.5], [now-12*base, 20.0], [now-8*base, 19.5], [now-4*base, 21.5], [now-1*base, 21]
      ]
   }
]
return msg;
```

Sensors, MyKPI and dynamic in this case
Multi Series without Time

- multi series with ordinal data

```json
{
  "metricId": "",
  "metricHighLevelType": "Dynamic",
  "metricName": "BatteryTemperatureGalaxyNote",
  "smField": "Gradi Centigradi",
  "metricValueUnit": "°C",
  "values": [
    [1, 19.5], [2, 20.0], [3, 20.5],
    [4, 8.5], [5, 19], [6, 18.5],
    ...........
    [50,5], [51, 21]
  ]
}
```

- You can set Staked via MoreOption

https://www.snap4city.org/dashboar
dSmartCity/view/index.php?iddasbo
ard=MjU3NQ==
IOT Application stressing Virtual Sensors Actuators concepts
Dashboard widgets can be classified in:

- **Virtual Sensors**
  - Those that produce the data
  
  *From IOT App on Dashboard*

- **Virtual Actuators**
  - Those that produce the data
  
  *From Dashboard to IOT App*

- **Virtual Sensors Actuators**
  - Those that produce/receive the data
  
  *From/to Dashboard to/from IOT App*
Sensors Actuators Allow to change the set up

Snap4City (C), January 2022
Sensors Actuators Allow to change the set up
IOT Application Integration with Synoptics
Dashboard on Browser

Internet

Storage and IOT App on cloud or on Premise

Case 1

WS

WS

MyKPI Sensor

Dashboard on Internet

Case 2

WS

WS

MyKPI Sensor

Dashboard on Internet

Case 3/4

WS Server

New Shared Variables

2500 Msg/s
Case 1 SVG ws³


10 WS messages per second
Case 2: Event Driven 100%


40 messages per second
WS Server
Case 3/4
New Shared Variables
2500 Msg/s

Dashboard on Browser
Internet
WS Server on cloud or on Premise

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MjgyNg==
Read more on

- **TC9.19: Custom Widgets / Synoptics controlled by IOT Applications**
- **Custom Synoptics and Widgets for Dashboards**
- **Scenario: 5G Enabled Water Cleaning Control**
- **Snap4Industry: Snap4City for Industry 4.0**
- **TC1.22: Create and configure a Snap4City SVG Custom Widget for real-time interaction**
DEMO
Section 2
Demo IOT Application exploiting Snap4City Dashboard
In this demo let's create an IoT Application that:

- send random values on Snap4city’s Dashboard
- create complex widget based on MyKPI e SURI
Nodes for flow

Generates an input for the other nodes. It can be repeated at predefined intervals, entered manually and of various types (timestamp, string, number, boolean, JSON etc).

Each message that enters the debug node is shown in the "debug" tab on the right of nodered (you can choose which part of the message to show).

Generates a random number. You can configure the number generation interval and the type (integer or float).

Display values in different modes on a dashboard. The node called single content accepts strings, numbers and html. The others only accept numbers.
Step 1

- Inject and Debug
- Connect
- Configure
- Deploy
- Click
- Observe
Step 2

- Random
- Connect
- Configure
- Deploy
- Click
- Observe

Snap4City (C), January 2022
Step 3

- Single content
- Connect
- Configure
- Deploy
- Click
- Observe
Nodes configuration
Nodes connections

- SingleContent - Random Value
- GaugeChart - Random Value
- SpeedMeter - Random Value
- Time Trend - Random Value

Each node is connected to ws://dashboard.km4city.org:8080/server.
Explaining: IOT Application Flow

• On Click or Every 15 minutes the `timestamp` node sends a message to the `random` node.

• When the message arrives, the `random` node generates a random number as output message.

• The Number can be sent to Different kinds of nodes to show it on NodeRed Dashboard.
Resulting Dashboard

DemoTrainingCourse2020

GaugeChart - Random Value
SingleContent - Random Value
SpeedMeter - Random Value

Time Trend - Random Value
Nodes for flow

Generates an input for the other nodes. It can be repeated at predefined intervals, entered manually and of various types (timestamp, string, number, boolean, JSON etc).

Each message that enters the debug node is shown in the "debug" tab on the right of nodered (you can choose which part of the message to show).

Search in around a certain point of the indicated service. It returns:
- servicesUri of all the services found,
- a GeoJSON containing a minimum of information about the services found, including the coordinates and the name of the service.

Retrieve the information about My KPIData saved on the Snap4city platform.

Display values in different modes on a dashboard. Check info of the node in the Node-RED tab.
Step 1

- Inject and Debug
- Connect
- Configure
- Deploy
- Click
- Observe
Step 2

- Service Search Within Circle
- Connect
- Configure
- Deploy
- Click
- Observe

Snap4City (C), January 2022
Step 1 Bis

- Inject and Debug
- Connect
- Configure
- Deploy
- Click
- Observe
Step 2 Bis

- Get My KPIData
- Connect
- Deploy
- Click
- Observe
Step 3

- Inject Node

- Configure with data of Weather Sensors and MyKPI retrieved at the previous steps

```json
[1] {  
  "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3166540",  
  "metricHighLevelType": "Sensor",  
  "metricName": "tusc_weather_sensor_ow_3166540",  
  "metricType": "airTemperature"  
},

9  
[9] {  
  "metricId": "http://www.disit.org/km4city/resource/tusc_weather_sensor_ow_3182522",  
  "metricHighLevelType": "Sensor",  
  "metricName": "tusc_weather_sensor_ow_3182522",  
  "metricType": "airTemperature"  
},

14  
[14] {  
  "metricId": "17857458",  
  "metricHighLevelType": "MyKPI",  
  "metricName": "Room 1",  
  "metricType": "Temperature"  
},

19  
[19] {  
  "metricId": "17857459",  
  "metricHighLevelType": "MyKPI",  
  "metricName": "Room 2",  
  "metricType": "Room Temperature"  
}
```
Step 4

• Bar Series
• Connect
• Configure
• Deploy
• Click
• Observe
Nodes connections
Resulting Dashboard
end DEMO

Section 2
Data Exchange and Distributed Computing on Multiple Snap4City Domains
• The new version of Snap4City Library on Node-RED support the management of Multiple Snap4City Platform Installations

• It is possible to:
  – Have in different Blocks, different registrations to different Snap4City Installations or Users
  – Get/Send data from/to a Snap4City Installations/Users and send/get to/from another
  – Have Multiple Brokers on multiple installations and users
  – Creating collaborative distributed processing that work and share data and processing in multiple platforms based on Snap4City or different.
Snap4City Multidomain Applications

Any Snap4City Installation
Different domain
Different user
Different auth./authoriz. System
Etc..

Any Snap4City Installation
Different domain
Different user
Different auth./authoriz. System
Etc..
Example on Dashboard multiple domains
IOT App = Node-RED + Snap4City

Data Analytics and WebScraping
IOT Application Listing, they can be

- Basic (white)
- Advanced (red)
- IOT Edge
  - Raspberry Pi
  - Android
  - Win/Linux
- Data Analytic (Plumber)
- Web Scraper (Portia)
IOT App vs Data Analytics in R-Studio
Data Analytics Dev. in R Studio and/or Tensor Flow

- Swagger
- SPARQL, FLINT
- Ontology Schema
- LOG.disit.org
- Knowledge Base, Km4City
- Big Data Store Facility
- Smart City API from Knowledge Base and other tools
- Creating Micro Services
- Using them into IOT Applications
- Saving Sharing Reusing
- Resource Manager
R Studio and Python algorithms are automatically transformed into MicroServices for your IOT Applications.

Data Analytics to MicroServices

S4CDataAnalytic
- descriptive statistics
- trend plot
- time series predictions
- machine learning predictions
- anomaly detection
- plumber data analytic
- python data analytic
Data Analytics to MicroServices with Plumber

How to configure the **plumber data analytic** node:

Relative Uri is the same of the R **@get** annotation:

````
#' @get /TuscanyHeatmap
```
Automatic_IDW_Heatmaps_Creation.R

#' @get /heatmapIDW
#' @serializer unboxedJSON
heatmapIDW <- function (city, long_min, long_max, lat_min, lat_max, epsgProjection, subnature, valueType, fromDateTime, toDateTime, heatmapName, colorMap) {
  ...
}

More Information

- **HOW TO: produce heatmaps, custom heatmaps on any data**
- **TC1.19: Creating and Exploit heatmaps for Dashboards and as reference data services**
PARAMETERS EXPLANATION

- **city** = name of the city related to the heatmap bbox (e.g., Rome, Florence, Helsinki, Antwerp etc.)
- **long_min**, **long_max**, **lat_min**, **lat_max** = heatmap bbox parameters (latitude and longitude coordinates)
- **epsgProjection** = UTM Projection related to the heatmap zone
- **subnature** = subnature of the sensor of interest (e.g., airQuality)
- **valueType** = single parameter or parameters array with the names of the measure of interest (the case of the array is related to multiple names for a single measure and NOT to multiple different measures - e.g., NO2, no2, airQualityNO2 etc.)
- **fromDateTime** = start date and time interval in timestamp format
- **toDateDateTime** = end date and time interval (for example 1-hour or 1-day or the timestamp format 2020-05-25T15:00:00)
- **heatmapName** = name of the heatmap
- **colorMap** = name of the color map associated to the type of measure of interest
IOT App vs Data Analytics in Python
Data Analytics Development in Python, Smart City API from Knowledge Base and other tools

- Swagger
- SPARQL, FLINT
- Ontology Schema
- LOG.disit.org

Knowledge Base, Km4City
Big Data Store Facility

Creating Micro Services
Using them into IOT Applications

Coding
Testing
Saving
Sharing
Reusing
Resource Manager

Snap4City (C), January 2022
Python process

- Develop Python code exploiting Flask calls
- Test on local for the Call
- Test on Cloud for API
- Deploy via IOT App
More information

• HOW TO: develop DataAnalytic in Python and manage them via IOT App
IOT App vs Web Scraping
Web Scraping from Portia

Knowledge Base, Km4City

My Scraping process

IOT App. Editor

Web Scraper PORTIA

Generating WEB Scraping

Sharing/saving reusing Scraping

Resource Manager

Snap4City (C), January 2022
Web scraping

• TC9.16 – Web Scraping to get data from web pages
Web Scraping

Snap4City (C), January 2022
IOT App self training
Self Training main path

• Please start a fully guided training cases:
  • HOW TO: create a Dashboard in Snap4City
  • HOW TO: add a device to the Snap4City Platform
  • HOW TO: add data sources to the Snap4City Platform
  • HOW TO: define privacy rules for personal data, produced by the end-users own device
  • HOW TO: Develop Smart Applications, Snap4City development Life Cycle
  • HOW TO: HLT vs Ingestion, and HLT vs Widgets
  • HOW TO: Develop an IOT Application for Data Ingestion
  • HOW TO: Upload data into Knowledge Base, ServiceMap (triple upload)
  • HOW TO: Create as set of Devices with BulkProcessing
  • HOW TO: Create an IOT Device Model
  • HOW TO: Create an IOT Device Instance from IOT Directory tool
IOT Applications vs Dashboards (self training)

- IOT Applications, realized by using Snap4City Node-RED and integrated with Snap4City Nodes/MicroServices block, can be behind dashboards to get data from them with Virtual Sensors and Actuators.
  - Dashboards may be connected to multiple IOT Applications and IOT devices
  - IOT Applications may be connected with multiple Dashboards and IOT devices
- A network of Dashboards, IOT Apps and IOT Dev and data is easily realized exchanging data via secure connections.
- see the following Training Cases
  - US2. Using and Creating Snap4City Applications with Dashboards
  - TC2.3 - List of MicroServices and the Help, for Final Users and Developers
  - TC2.4 - The daisy of MicroServices for Snap4City Dashboard and IOT App
  - TC2.28 - Snap4City MicroServices for Snap4City platform management from IOT Applications, feature of reflection
IOT Applications vs Dashboards (self training)

- see the following Training Cases
  - **US2.** Using and Creating Snap4City Applications with Dashboards
  - **TC2.3** - List of MicroServices and the Help, for Final Users and Developers
  - **TC2.4** - The daisy of MicroServices for Snap4City Dashboard and IOT App
  - **TC2.28** - Snap4City MicroServices for Snap4City platform management from IOT Applications, feature of reflection
  - **TC2.24** - IOT Applications developed exploiting MicroServices, also supporting GDPR, real time, data sharing, etc.
  - **US9.** Creating Snap4City IOT Applications, different formats, protocols, brokers, communications
  - **TC6.8** - ETL processes for data transformation, and exploiting MicroServices/API/RestCall
  - **TC2.13** - Import of any new Block/MicroService or library of MicroServices into IOT Application Builder tools
Self Training articles


• See also courses in ITALIANO:  https://www.snap4city.org/485
MicroServices Suite for Smart City

- Badii, C.; Bellini, P.; Difino, A.; Nesi, P.; Pantaleo, G.; Paolucci, M. MicroServices Suite for Smart City Applications.
- *Sensors* 2019, 19, 4798.
- [https://www.mdpi.com/1424-8220/19/21/4798/pdf](https://www.mdpi.com/1424-8220/19/21/4798/pdf)
IOT App = Node-RED + Snap4City examples
A public operator (Road Operator) on the field, like a policeman or a public transport driver, notifies to a control room operator (City Operator) a critical event in the city.

The notification includes the reporting in real time of the event position, the number of involved people and the gravity of the event.
Alerting about critical events involving people in a specific area

A public operator (Road Operator) on the field, like a policeman or a public transport driver, notifies to a control room operator (City Operator) a critical event in the city.

The notification includes the reporting in real time of the event position, the number of involved people and the gravity of the event.
Check which route is less polluted

In this example, microServices retrieve information from the Smart City storage and info to create a dashboard that tells the user which is the less polluted path at a precise moment to go jogging.

If predictive data are available, it can work on predictions.

Exploitation of MicroApplications
Check which route is less polluted

In this example, microServices retrieve information from the Smart City storage and info to create a dashboard that tells the user which is the less polluted path at a precise moment to go jogging.

If predictive data are available, it can work on predictions.
Controlling Personal Mobile PAX Counter

In this example, the interaction with IOT Devices counting people by using Wi-Fi and Bluetooth sniffing in its vicinity (according to GDPR)
In this example, the interaction with IOT Devices counting people by using Wi-Fi and Bluetooth sniffing in its vicinity (according to GDPR).
What we are going to do now!

• Create a Simple IOT Application (Demo)
• Production of IOT Application (Exercitation)
• Data Processing with IOT Application (Demo)
• Processing Data with IOT Applications (Exercitation)
Start DEMO
Section 3
Start DEMO

Section 3
Create a Simple IOT Application (DEMO)
Demo of Simple IOT Application

In this demo let's create an IOT Application that:

- reads a realtime value of a service and
- publishes it on a dashboard
- sends email to someone
Nodes for flow

Generates an input for the other nodes. It can be repeated at predefined intervals, entered manually and of various types (string, number, Boolean, json etc.)

Requests detailed information for a specific service on the platform (such as a car park, hotel, etc.)

Executes a Javascript code once the input message is received

Transforms the incoming message into a JSON

Display values in different modalities on a Dashboard (or on different Dash)
   The node called single content accepts strings, numbers and html.
   The others only accept numbers.

Send an email to the desired recipient. You must enter the username and password of an active email.
Step 1

• Inject and Debug

• Connect

• Deploy

• Click and Observe

• Play with results
Step 2

• Service Info
• Connect
• Configure
• Deploy
• Click and Observe
• Play with results

Copy the path

Copy the value

http://www.disit.org/km4city/resource/CarParkPieracciniMeyer
Step 3

- Function
- Connect
- Configure
- Deploy
- Click and Observe
- Play with results

msg.payload = msg.payload.realtime.results.bindings[0].freeParkingLots.value
Step 4

- JSON
- Connect
- Deploy
- Click and Observe
- Play with results
Step 5

- Single content
- Connect
- Configure
- Deploy
- Click and Observe
- Play with results
Step 6

- Email
- Connect
- Configure
- Deploy
- Click and Observe
- Play with results

Change if not GMAIL
Nodes configuration

Payload: timestamp
Topic:
Repeat: interval
every 15 minutes
Inject once at start?

Service info
Name:
Service Uri: http://www.disit.org/km4city/resource/CarParkPieracciniMeyer
Language: Italian

Function
Name: Get Free Parking Lots
Function:
1 msg.payload = msg.payload.realtime.results.
2 bindings[8].freeParkingLots.value
3 return msg;

Dashboard
Name: BasicDemo023Luglio
Create New
Widget
Name: SingleContent - PieracciniMeyer
Edit Dashboard View Dashboard
Nodes connections

- SingleContent - PieracciniMeyer
  - connected to ws://dashboard.km4city.org:8080/server

- GaugeChart - PieracciniMeyer
  - connected to ws://dashboard.km4city.org:8080/server

- SpeedMeter - Pieraccini Meyer
  - connected to ws://dashboard.km4city.org:8080/server

- Time Trend - Pieraccini Meyer
  - connected to ws://dashboard.km4city.org:8080/server
Explaining: IOT Application Flow

• On Click or Every 15 minutes the timestamp node sends a message to the service-info node.

• When the message arrives, a request is sent to get details of the service URI entered in the configuration, in this case the Pieraccini Meyer car park.

• The details are sent to the node named ”Get Free Parking Lots”, which recovers the value of the current free places and ignores all the other data received in response.
  • The values in output of node Get Free Parking Lots is a string.

• THUS ! node json may transform it into a number (for those who know JavaScript could be used function parseInt() inside the function node). Then a number has been obtained!

• The Number can be sent to Different kinds of nodes to show it on Dashboards Widgets.
end DEMO

Section 3
Production of IOT Applications
Exercitation
IOT Application Exercitation

Goal:

Create an IOT App (flow) that reads a value from a service (for example the parking lot seen in the previous demo)

serviceUri: http://www.disit.org/km4city/resource/CarParkPieracciniMeyer

and:

based on a certain threshold sends a different message on the dashboard. For example, Almost Full Parking or Free Parking. OR Send to you an email 😊 !

You have 15 Minutes!
Ex1: Your NickName: ........................................
One Possible Solution
Nodes configuration

Free Park
- **Name**: Free Park
- **Function**:
  ```javascript
  msg.payload = {
  1: {
    msg.payload = "<b style='color: green'>Free</b> " + msg.payload + "</b>"
  };
  return msg;
  ```

Busy Park
- **Name**: Busy Park
- **Function**:
  ```javascript
  msg.payload = {
  1: {
    msg.payload = "<b style='color: red'>Full</b> " + msg.payload + "</b>"
  };
  return msg;
  ```
Resulting Dashboard

Threshold

Free 207

Start DEMO
Section 4
Start DEMO
Section 4
Data Processing with IOT Application (DEMO)
Example of more Complex IOT Application

In this demo let's create an IoT Application that:

• reads a realtime values from a list of services,
• makes the sum of the value and
• publish the result on a dashboard
Nodes for flow 1/2

Generates an input for the other nodes. It can be repeated at predefined intervals, entered manually and of various types (string, number, Boolean, json etc)

Requests detailed information for a specific service on the platform (such as a car park, hotel, etc.)

Search in around a certain point of the indicated service. It returns:
- servicesUri of all the services found,
- a GeoJSON containing a minimum of information about the services found, including the coordinates and the name of the service.

Executes Javascript code. For example, exploiting data arrived on input message and producing an output message in JSON

Display values in different modes on a dashboard. The node called single content accepts strings, numbers and html. The others only accept numbers.
Nodes for flow 2/2

Divides the input message into multiple messages as indicated in the configuration.
If you have an array at the input, you can configure it to send each element of the array individually at the output.

Treads the input message on possible different outputs based on a comparison made on the input message.

Operates in reverse order to the split. Joins the incoming messages in the mode indicated in the configuration.
Nodes configuration 1/2

**Inject**
- **Payload**: timestamp
- **Topic**: 
- **Repeat**: interval every 15 minutes
- **Inject once at start?**

**Array**
- **Split using**: Fixed length of 1

**service info**
- **Name**
- **ServiceUrl**: http://
- **Language**

**Dashboard**
- **Name**: TotalFreePark
- **Widget**
  - **Name**: Gauge - TotalFreePark

**Function**
```
var sum = 0;
for (var i = 0; i < msg.payload.length; i++){
    sum = sum + parseInt(msg.payload[i].realtime.results.bindings[0].freeParkinglots.value);
}
msg.payload = sum;
return msg;
```
Nodes configuration 2/2
Nodes connections

- **Timestamp**
- **Service search within circle**
- **Split**
- **Service info**
- **Switch**
- **Join**
- **Sum Of Free Park**

- **Gauge - Total Free Park**
  - Connected to ws://dashboard.km4city.org:8080/server

- **Single Content - Total Free Park**
  - Connected to ws://dashboard.km4city.org:8080/server

- **Speedometer - Total Free Park**
  - Connected to ws://dashboard.km4city.org:8080/server

- **Time Trend - Total Free Park**
  - Connected to ws://dashboard.km4city.org:8080/server
Nodes explanation 1/2

• Every 15 minutes the **timestamp** node sends a message to the **service-search-within-circle node**.
  • When this message arrives, a request is sent to find all the car parks in the search area entered in configuration
• The first output of the **service-search-within-circle** node returns an array containing all the URI services of the car parks found. On such array we effect a **split** so that in input to **service-info** all the services URI arrive as distinct messages in a sequence.
• The configuration of the **service-info** node has not been filled because the URI service comes from the incoming message and is considered that URI service for retrieving service details.
• The **switch** and **join** nodes are used respectively to filter the results eliminating those parking lots that have no value in realtime (because for example that parking lot has no sensor) and bring together the various messages in a single array.
• On this array, node **Sum of Free Park** the perform the sum of the free places of all Florence parking and sent to the value to nodes representing Dashboard Widgets.
TotalFreePark

Gauge - TotalFreePark

Speedometer - TotalFreePark

SingleContent - TotalFreePark

TimeTrend - TotalFreePark

end DEMO
Section 4
Processing data with IOT Applications (Exercitation)
Average IoT Application

Create an IOT Application / flow that:
- reads a value from a list of service, for example the car parks in the Florence City Area, as seen in previous demo and
- calculates the average of Free Parking Lots and
- sends the value on a dashboard with the four possible nodes seen in the demo.

Execution Time: 20 Minutes
### Ex2: Your NickName:

<table>
<thead>
<tr>
<th>Input</th>
<th>Function</th>
<th>S4C Search</th>
<th>S4CDashboard</th>
<th>S4CKPIData</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>function</td>
<td></td>
<td>get my kpidata</td>
</tr>
<tr>
<td>inject</td>
<td></td>
<td>service search near marker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>delay</td>
<td></td>
<td>service search within circle</td>
<td></td>
<td>get my kpidata</td>
</tr>
<tr>
<td>debug</td>
<td></td>
<td>service search within polygon</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>service search along path</td>
<td></td>
<td>get public kpidata</td>
</tr>
<tr>
<td></td>
<td></td>
<td>service info</td>
<td></td>
<td>values</td>
</tr>
<tr>
<td>http request</td>
<td></td>
<td>full text search near marker</td>
<td></td>
<td>get delegated</td>
</tr>
<tr>
<td>top request</td>
<td></td>
<td>full text search within circle</td>
<td></td>
<td>kpidata values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>full text search within polygon</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>full text search along path</td>
<td></td>
<td>save my kpidata</td>
</tr>
<tr>
<td></td>
<td></td>
<td>full text search usr</td>
<td></td>
<td>values</td>
</tr>
</tbody>
</table>
One Possible Solution
Nodes configuration 1/2

- **Inject**
  - Payload: timestamp
  - Topic
  - Repeat: interval, every 15 minutes
  - Check: Inject once at start?

- **Array**
  - Split using: Fixed length of 1

- **service info**
  - Name
  - ServiceUrl: http://
  - Language

- **Dashboard**
  - Name: TotalFreePark
  - Widget: Gauge - TotalFreePark
  - Name: Gauge Of Free Park
  - Function:
    ```javascript
    var sum = 0;
    for (var i = 0; i < msg.payload.length; i++) {
        sum = sum + parseInt(msg.payload[i].realtime.results.bindings[0].freeParkingLots.value);
    }
    msg.payload = parseInt(sum/msg.payload.length);
    return msg;
    ```

Snap4City (C), January 2022
Nodes configuration 2/2

- Service search within circle
- Switch
  - Name: Name
  - Property: msg.payload.realtime.results
  - Is not null
- Join
  - Mode: manual
  - Combine each: msg.payload
  - To create: an Array
  - Send the message:
    - After a number of message parts
    - After a timeout following the first message
Resulting Dashboard

https://main.snap4city.org/view/index.php?iddasboard=MTk2Mg==
Self Training articles


• See also courses in ITALIANO:  https://www.snap4city.org/485
Integration of External Services into IOT Applications
External Services

• Any service having some API (rest, WS, web services, etc..) can be called and exploited from IOT App. Also for example machine learning tool on AWS, MicroSoft cloud, Google Cloud. They can access to Snap4City Data by using Smart City API, ASCAPI

• Any REST Call can be transformed into a Block/Node for Node-red and thus exploited in IOT Apps.

• Example of integration of IOT App Snap4City with:
  – Workflow management system for Ticketing and Incident Maintenance
  – Twitter Vigilance, data collection
  – CKAN for Open Data gathering and publication
  – Video Wall for Control Room
  – Copernicus, etc.
Integration with Ticketing Systems
Workflows, Incident Management
Snap4City Maintenance Solution

- **OpenMaint** open source solution for property & facility management which is a BPM;
  - Inventory of industry assets (movable, logistics, equipments, etc.)
  - Tickets management for corrective maintenance
  - User management with different levels of access
  - BIM Server integrated with OpenMaint

- **Snap4City OpenMaint Extension**
  - **Extended API** developed by Snap4City
    - Create new tickets
    - Manage steps, workflow
    - Collecting feedbacks and results from teams
    - Manage all phases of the workflow on the fields via IOT Apps and logics
    - The integration if via API and MicroServices into IOT App.
  - **MicroServices** integrated with Snap4City via IOT Applications

- **Business Intelligence** which is the **Snap4City tool based on** OpenDistro for ElasticSearch: which work on top of the database of tickets collected on OpenMaint

- **BIMServer** integration with Snap4City Dashboards;
Example of Integrated workflow

Consumptions/productions

Events/actions

Business Intelligence Maintenance

Dashboards and actions

OpenMaint: BPM Workflow management, team assignment, material control, ...

IOT App, Data event firing, event detection and firing

Critical event management

Snap4City (C), January 2022
Integration with Ticketing Systems Workflow

• Snap4City is integrated with OpenMaint Ticketing system. An Open Source solution for ticketing and workflow management, incident management.

• Any ticketing systems can be integrated with Snap4City, by means of IOT Applications and Dashboards

• https://www.snap4city.org/597
Assets inventory and management
Solution for Asset Management and Maintenance

• Inventory of industry assets (movable, logistics, equipments, etc.)
• Tickets management for corrective maintenance
• Reports and Dashboards
• Predictive maintenance and Early Warning support via analytics
• Business Intelligence support
• User management with different levels of access
OpenMaint Ticketing System

- Define the workflow
- Activate events
- Manage steps
- Collecting feedbacks and results from teams

- Activities
- Teams
- Phases
- Authorization
- Checking points
- Events and conditions
  ...

Snap4City (C), January 2022
• Snap4City can
  • Create new tickets
  • Manage steps, workflow
  • Collecting feedbacks and results from teams
  • Manage all phases of the workflow on the fields via IOT Apps and logics
  • The integration if via API and MicroServices into IOT App.
Business Intelligence
BIM view of the Altair Chemical Plant

BIM Integration Dashboard


Snap4City (C), January 2022
Snap4BIM: from 3D model to real time data
Integration with Twitter Vigilance
• A separate tool of DISIT lab: provided in different versions
  – Described into Data Analytic part of the training course
• It can be used to:
  – Collect and Monitor Twitter data
  – Perform Multilingual processing: English, France, Italia, etc.
  – Estimated in real-time metrics: volume, sentiment, ratio,
  – Provide data into the smart city and thus alerting and firing
  – Compute predictions
  – Set up Early Warning systems

• Snap4City integration is done via API and MicroServices into IOT App.
IOT Applications Development

- MicroServices collections
- My IOT Applications
- IOT App. Editor
- Generating IOT App With Dashboard
- Sharing/saving reusing IOT App

ServiceMap Discovery
Dashboard Collection, Editor and Wizard
Knowledge Base, Km4City

Snap4City (C), January 2022
Twitter Vigilance

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=Mjc3NQ==

Snap4City (C), January 2022
• Snap4City integration is done via API and MicroServices into IOT App.
• This is an example of Dynamic Widget Data production for MultiSeries
• Each country block produce a vector and the vectors are joined and sent to the Multiserie Widget
Further reading on Social Media

- TC2.22 - Exploiting Twitter Vigilance as External Service, in Dashboard, and as RestCall as MicroService in IOT applications
- TC2.21 - IOT Applications with Social Media Actions, and cultural scenarious
- External Services
- TC2.21 - IOT Applications with Social Media Actions, and cultural scenarious
Integration with CKAN: automated data set (i) ingestion and (ii) production via IOT App
Snap4City vs CKAN

Automatize:
- Import data from CKAN to Snap4City
- Upload Public Data from Snap4City to CKAN
- Data Harvesting
- Dashboards and Mobile/Web Apps creation
Some IOT App segments

Almost all the calls to CKAN are quite similar
• **TC9.17 – CKAN vs Snap4City Integration and Interaction**

  – automating the *Read of a Dataset Info from CKAN*
  – automating the *Read of a Resource info from CKAN*
  – automating the *Creation of a Dataset on CKAN*
  – automating the *Creation of a static Resource in CKAN*
  – automating the *Creation of a dynamic Resource in CKAN*
  – automating the *Sending of a json to CKAN from a query to Snap4City to perform any other action on the Smart City*

• **Data Set Manager: Data Gate / CKAN federated**
Automated production of MicroService for IOT App from External REST CALL API
You can **save/consolidate your rest API** transforming it in a MicroService usable for many colleagues into IOT Applications:

- TC2.25- Registering external MicroService calling RestCall services, using it on IOT applications [https://www.snap4city.org/129](https://www.snap4city.org/129)

**IF** your REST API is going to use credentials as username and password, we suggest you to save them into MyPersonalData of Snap4City

- so that the code will not provide clear credentials and you can update from user interface on your personal data profile.
- The IOT App can retrieve the Username and Password at the moment in which they are used with the security shield of Snap4City
External REST Call API vs MicroServices

- Each REST Call API can automatically be transformed into a MicroService for the IOT Applications

https://www.snap4city.org/129

Snap4City (C), January 2022
• Each Rest Call API can be automatically transformed into a MicroService for the IOT Applications.
**Edit MicroService Call: call and help editing**

---

### Edit MicroService: Antwerp cameras location.zip

- **Nature:** Transfer service and renting
- **Sub Nature:** Monitoring camera
- **Licence:** Public
- **Description:** Antwerp cameras location from A Open Data
- **Select Image:** [Open file] Nessun file selezionato
- **Method:** GET
- **Do you want create a Microservice with Authentication?** No

**Help:**

- **Source**

**Styles**

**Description of microservice**

The service gives the camera location (lat, lon)

**Inputs**

**Microservice input description:**

- **No Parameter**

**Outputs**

- **json**

**Details**

More details here: [Visit Details] https://opendata.antwerpen.be/datasets/kaart

---

**Formal definition**

---

**HELP**
Usage of the MicroService from IOT App
Integration with Telegram: SnapBot solution
SnapBot provides real-time smart city services to Telegram users, geolocalized, when you like, what you like.

active on Tuscany in all provinces and cities according to the data accessible on [https://www.snap4city.org](https://www.snap4city.org)

Services on:
- Public Transport (more than 10 different operators),
- bike sharing, parking lots,
- traffic flow, weather warnings,
- Air quality, pollutant,
- find your location, etc.
SnapBot

Qualità dell’aria rilevata dal sensore più vicino alla posizione:
- Temperatura: 8.10 °C
- Umidità: 97.50%
- CO: 0.3 μg/m³
- CO₂: 499.0 μg/m³
- NO: NaN μg/m³
- NO₂: 56.1 μg/m³
- O₃: 20.9 μg/m³
- PM10: 13.8 μg/m³
- PM2.5: 12.2 μg/m³

Ho trovato 6 linee vicino a te:

24 - ATAF&LINEA
Grassina → Bagno A Ripoli Robinson

49 - ATAF&LINEA
Grassina 02 → Bagno A Ripoli Robinson

48 - ATAF&LINEA
Il Roseto 01 → Bagno A Ripoli Robinson
IOT App of SnapBot: OneShot Services

Exploiting Node-RED Snap4City MicroServices
IOT Network Management and Control
IOT Basic

IOT Device

IOT Broker/ IOT Gateway

Subscribe

IOT Application

Publish

IOT Broker/ IOT Gateway

Sensors

- Sensors are programmed to send data (i) periodically, or (ii) when a relevant change occurs in the sensor value, or (iii) when events occur (for example a change of status of something), etc.
- Actuator perform some action on the field: change of status, reset, turn on something, change setting value, etc.

IOT Apps are

- data driven functional programs for data transformation.
- can subscribe to some IOT Brokers to receive data in Push from a specific IOT Device (sensor)
- can publish some message toward some IOT Device (Actuator), passing via an IOT Broker.
- Can be used to create ADAPTERs of any kind

Continuous lines are messages via TCP/IP
Dashed lines are message via some radio channel (Lora, BT, Wi-Fi, ...)
IOT Brokers and IOT Gateway can be distinct servers
IOT Brokers can be on cloud
IOT Gateway performs the SW update, the business management, access in Push and Pull
- IOT Brokers can be connected each other
- Adapters / Wrappers transcode one message from one protocol to another

- IOT Edge may include IOT Apps
- Missing knowledge about the semantic of IOT devices
- Lack of capability for IOT Discovery: value type, location, etc., which could be used by IOT App
- Lack of Storage of data values over time
Definitions

• IOT Edge may include IOT Applications
• Missing knowledge about the semantic of IOT devices
• Lack of capability for IOT Discovery: value type, location, etc., which could be used by IOT App
• Lack of Storage of data values over time
Federation of Snap4City vs IOT ORION Broker

- **IoT Agent**
- **IoT Orion Broker**
  - Crate-DB: 4200:4200
  - MongoDB: 27017:27017
  - Quantum Leap: 8668:8668

**Dashboard Builder**

**Federation**
- **SSM2ORION**
- **SUPER**
- **NGSI**

**Hybrid Solutions**

**Snap4City Solutions**
IOT/IOE Protocols

Communication Patterns

- MQTT
- HTTP(s)
- AMQP
- COAP
- NGSI
- OneM2M
- WebSockets
- .......
- Etc.
Note on Communication patterns

• Not all Communication Patterns are supported by all Protocols
• Protocols implement Patterns, + formats, + sequences, etc.
• They are referred at level of communications
  – IOT Device ← → IOT Gateway ← → IOT Broker
• IOT Protocols mostly used at level of IP are:
  – NGSI V1/2, MQTT, COAP, AMQP, OneM2M, WS, ModBUS,
• Radio protocols are: Lora, ZigBee, 3G, Wi-Fi, etc.
• Formats: JSON, Geo-JSON, Linked Data, XML, CSV,
IOT Networks on Snap4City
SNAP4CITY

100% OPEN SOURCE

APPLIANCES CONTAINERS
- LOCAL GOVERN
- STAKEHOLDERS
- CITY USERS
- IN-HOUSE
- ENERGY OPERATORS
- MOBILITY OPERATORS
- COMMERCIAL OPERATORS
- SECURITY OPERATORS
- INDUSTRIES
- RESEARCHERS
- START-UPS
- ASSOCIATIONS

URBAN PLATFORM: SMART CITY IOT AS A SERVICE AND ON PREMISE

IOT APPLICATIONS - INSTANT APPS
- DATA DRIVEN APPLICATIONS
- REAL TIME PROCESSING
- BATCH PROCESSING
- ANY PROTOCOL & FORMAT

DASHBOARDS & APPLICATIONS
- CONTROL ROOM
- SITUATION ROOM
- OPERATOR DASHBOARDS
- BUSINESS INTELLIGENCE
- WHAT-IF ANALYSIS
- DECISION SUPPORT
- SIMULATIONS
- RISK ANALYSIS
- RESILIENCE ANALYSIS

MOBILE & WEB APPLICATIONS
- DEVELOPMENT KIT
- SUGGESTIONS
- MOBILE APPS
- MONITORING PANELS
- PLATFORM UTILITIES
- READY TO USE SMART APPLICATIONS

MICROSERVICES & ADVANCED SMART CITY API
- DATA ANALYTICS TOOLS - MICRO-APPLICATIONS
- PREDICTIONS
- ANOMALY DETECTION
- WHAT-IF ANALYSIS
- TRAFFIC FLOW RECONSTRUCTION
- ORIGIN-DESTINATION MATRICES
- SOCIAL MEDIA ANALYSIS
- OFFER VS DEMAND ANALYSIS
- ENVIRONMENTAL DATA ANALYSIS
- AND PREDICTIONS
- REAL TIME HEATMAPS
- ROUTING
- ALERTING
- EARLY WARNING
- PERSONAL AND VIRTUAL ASSISTANTS
- SMART SOLUTIONS
- SMART SHARING
- PARTICIPATORY

IOT MNG - DATA MNG - DATA INSPECTOR - PROCESS MNG - USER ENGAGEMENT - GDPR MNG ...

KM4CITY DATA AGGREGAT KNOWLEDGE BASE - EXPERT SYSTEM OF THE CITY – BIG DATA STORE

OIS CITY UTILITIES OPEN DATA LEGACY & EXTERNAL SERVICES PERSONAL DATA IOT / IOE BROKERS KPI INDUSTRY 4.0 SOCIAL MEDIA
Standards and Interoperability (2021)

Snap4City vs Formats

• Snap4City is capable to ingest and work with any format:
  – Data **exchange**: JSON, GeoJSON, XML, HTML, HTML5, DATEX, GTFS, binary, etc.
  – **Table**: CSV, XLSX, XLS, database, ...
  – Any **archive** file format: zip, rar, 7z, tgz, ...
  – Any **image** format: png, gif, tiff, ico, jpg, ...
  – Any **video** format: mp4, avi, mov, ...

• Search the format you need to cope on the search box of Snap4City portal!
IOT Network Manager vs Final User

- Network of IOT Brokers
- IOT Directory
- IOT Network Manager
- My IOT Device
- IOT Application
- Dashboard Wizard
- ServiceMap
- Knowledge Base, Km4City

Key Features:
- Knowledge and Storage
- Data from the Field and City
- Discovering
- Registering
- Browsing
- Exploit
- Prepare
- Deep Manage
- Register
- Final user Manager

Snap4City (C), January 2022
Main Features of the Snap4City IOT Directory:

- **IOT Directory is a technology of Snap4City ONLY**
- **Register IOT Brokers**
  - Different kind of Brokers, different kinds of authentications and protocols
  - Registered IOT Orion Brokers can be queried for collecting their managed devices (typically for External IOT Brokers), so that those IOT Devices are registered
  - IOT Brokers/Gateways are registered on NIFI to send messages into the Data Shadow, automatically
- **Register IOT Devices**: singularly or at groups (in Bulk)
  - Registration can be custom or based on IOT Device Model
  - IOT Edge are registered as IOT Devices as well
  - Registered IOT Devices are saved into local DataBase and into the Knowledge Base
- **Provide support for security aspects**:
  - Generation of Certificates, Keys, etc., according to the model
  - Collection of keys when IOT devices are on some IOT Gateway or Second Level IOT Broker.
- **Manage Ownership and Delegation** for
  - IOT Brokers, IOT Devices, IOT Device Values also called Sensors/actuators, IOT Device Models
# IOT Directory Features vs Users Roles (10/21)

<table>
<thead>
<tr>
<th>Entities</th>
<th>what</th>
<th>By using IOT Directory and:</th>
<th>Manager</th>
<th>AreaManager</th>
<th>ToolAdmin/RootAdmin</th>
<th>IOT App microservices</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOT Sensor/Actuator</td>
<td>Browse, use</td>
<td>Several Tools</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Delegate</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Discovery</td>
<td>KB, API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td>IOT Devices</td>
<td>Browse, use</td>
<td>Several Tools</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes (use)</td>
</tr>
<tr>
<td></td>
<td>Create, change, delete</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Register in Bulk</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Delegate, Change Owner</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Discovery</td>
<td>KB, API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td>IOT Device Model</td>
<td>Browse, Use</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>(Yes)</td>
</tr>
<tr>
<td></td>
<td>Create, change, delete</td>
<td></td>
<td>X</td>
<td>X</td>
<td>(Yes)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>delegate, change ownership</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOT Broker</td>
<td>Browse, use</td>
<td>use</td>
<td></td>
<td>Browse, use</td>
<td>X</td>
<td>Yes (use)</td>
</tr>
<tr>
<td></td>
<td>Register/change/Delete</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deploy Orion Broker</td>
<td></td>
<td></td>
<td></td>
<td>ToolAdmin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Delegate</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Periodic Update</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
In which case you are?

https://www.snap4city.org/drupal/node/474

**Case A1.2**

IOT Device

IOT Broker

a) Registered IOT Device on Broker

IOT Broker

i) Registered IOT Broker on Snap4City

**Case B1**

IOT Device

IOT Broker

i) Registered IOT Broker on Snap4City

**Case B2**

IOT Device

IOT Broker

a) Registered IOT Device on Broker

IOT Broker

i) Registered IOT Broker on Snap4City

ii) Registered IOT Device on Snap
A. Already registered on an IOT Broker of your/city organization. In this case, who provided you the IOT Device may have provided also at least: an IOT Device Identifier, a description of the data produced by the Device, the protocol kind and the IOT Broker in which it is registered, etc. For example: device ID: es286481295, temperature and humidity, NGSI, the “orionFinland” IOT Broker on https://ngsi.fvh.fi or the “Antwerp” IOT Broker on https://ext-api-gw-p.antwerpen.be. In order to exploit the data of your IOT Device in the Platform, you

1. have an **IOT Device which is registered** on an **IOT Broker** (for example, you received with the IOT Device the name of the IOT Broker) that is registered to Snap4City. You have two cases:
   1. the IOT Device has been **already registered** on Snap4City by the organization/city. This case can be verified by using the steps described in:
      – See in this case: HOW TO: verify if an IOT device is registered and accessible for me.
      – if the IOT device is registered please note that you are in case A3, if not, go at case A1.2
   2. the IOT Device is **not yet registered** on Snap4City (for example when an IOT Broker is managed by a third organization for security aspects, for example the FHV or DIGIPOLIS, or IMEC, ...), they given to your the device to test on different platforms.
      – See in this case: HOW TO: Add an IOT Device on Snap4City platform that is already registered on an external IOT Broker

2. need to access IOT Device data **without registering the IOT Device in the Snap4City platform and Broker**, you need to know some configuration parameter of the IOT Broker and IOT Device, and Snap4City IOT App can get data directly from the IOT broker of your device without the need of having the Broker officially registered on snap4City.
   – See in this case: HOW TO: add IOT Device data source from external broker to the platform.

2. have an **IOT Device which is already registered to an internal Snap4City IOT Broker** (a IOT Broker managed by Snap4City for security aspects). In this case, the IOT Device and corresponding data are immediately accessible, and you can find them into the list of your data in the Data Inspector view, for Dashboards, etc., go in the Data Inspector to search your data by GPS location, name, nature as you like.
A. Already registered on an IOT Broker ... see above

B. Not registered to an IOT Broker. In this case, you need to know, at least, how the IOT Device works and how it can be internally configured to communicate with an IOT Broker: to authenticate, register, etc. So that you need to know: an IOT Device Identifier, a description of the data produced by the device, the protocol, etc. For example: device ID: 286481295, temperature and humidity, NGSI with basic authentication,

1. In this case, the first step is to register the IOT Device to an IOT Broker. Snap4City offers you a number of Snap4City IOT Brokers compliant with different protocols to which you can connect your device. To this end, please follow this tutorial:
   - See in this case: HOW TO: Add an IOT Device on Snap4City platform by registering it on an Internal Snap4City IOT Broker

2. in alternative you can find some other brokers in your area according to the protocol of your device.

3. Once registered the IOT Device to an IOT Broker please restart from case (A); if you registered with a Snap4City IOT Broker it will be easy an (A2).
How to setup and IOT Data Stream

Managers/AreaManagers:
1. Register the IOT Broker you want to use.
   – If you do not have one, you can ask one to Snap4City
2. Register the IOT Device you want to use.
   – If it is only one Device to reg, you can do it manually,
   – if they are many, we suggest you to create an IOT Device Model, then register the device (only AreaManagers)
3. Use IT

Administrators:
1. Register the IOT Broker you want to use, or use one already registered.
   – If the IOT Orion Broker has IOT Devices registered in you can use the procedure for automated registration (from your Broker to the IOT Directory and KB), with rule for transformation, etc.
   – If not see points 2 and/or 3
2. Register a single IOT Device manually
3. Register a group of IOT Devices
   – create a IOT Device Model
   – Create a CVS file for Registering devices in Bulk
4. Use IT
Data Ingestion Strategy
Snap4City Architecture vs Data Ingestion

Data Sources, External Services

Data Sources, Brokers, External Services

Data Ingestion, aggregation, regularization, reconcile:

Knowledge base
Semantic reasoners

Indexing and aggregating
OpenDistro (Elastic search)

Search and Query,
Smart City API
Facet, semantic search

Data Analytics, Simulations
R, Tensor Flow, Python, ….

IOT Applications, Business Logic
Node-RED + Snap4City MicroServices

Inform, announce, Act!, warning, alarms, What-IF, ..
POI Loader

- To help you to upload POI data in short/zero time: https://www.snap4city.org/731
  - Start from Excel Files, they should be formatted some how or well formatted according to our guidelines (model provided)
  - Custom upload for each Organization is possible on the provided IOT App

- To enable you to
  - create dashboards from them according to different views and nature
Short cut Data Ingestion from Excel file

1) Upload the file on **Data Table Loader**
2) Follows the instructions and guidelines -- the dirty work will be done in a Snap -- wait!
3) See data on your Data Inspector 😊
4) Use Data Into Dashboards 😊
Checking data ingestion results

- Data Inspector
- ServiceMap, SCAPI
  - LOG / LOD viewer
  - Super Service Map
- IOT Directory
- SCAPI: Swagger
- IOT Broker
- Data Inspector
- ServiceMap, SCAPI
- My Data Dashboard (Kibana), DevDash
- OpenDistro x Elastic Search

Some functionalities are limited to certain roles
On the Flow Diagram

• **Static** (unified model for multidomain indexing)
  – Geodata, Open Data as POI, Data Bases, records, etc.
  – They change over time sporadically

• **Models** (Registration of IOT Device Models, IOT Devices, Brokers)
  – Registration of entities with their metadata and data structures
    • Brokers, Devices, structures of real time data, machine models for IOT Industry 4.0, sensor models/structure, etc.

• **Time Series** (dynamic data)
  – Any instance of IOT Devices over time and space with any kind of entity relationship each other and with other city entities
Snap4city Data Ingestion Flow Diagram

- **Static or quasi Static Data:** POI, etc.
- **IOT Broker Registration**
- **IOT Device Model Reg.**
- **IOT Device Registration**
- **Real Time data in Pull** Any gateway, server
- **IOT App**
- **IOT Directory**
- **IOT Orion Broker**
- **Knowledge Base Semantic Reasoners**
- **Indexing and Aggregating**
  - **NIFI, OpenDistro ElasticSearch**

**IOT Device**
- **push**

**IOT Device Reg.**

**IOT Orion Broker**
- **push**

**IOT App Adapter**
- **Automatic**

**IOT Devices**

**IOT Brokers**

**SURI Link**

**Information, File**
- **IOT Apps**
- **Snap4City Tools**
- **IOT Device/Gateway**
• The IOT Orion Brokers can be feed by means of
  – IOT App of Snap4City (to implement Agents and/or Adapters)
  – IOT Agents and/or NGSI Adapters by FiWare for different protocols
  – IOT Brokers of any kind, different protocols and producers, also as Gateways, and they can be located on premise and/or on any cloud

• IOT App, IOT Agents, Adapters can
  – be on IOT Edge
  – be implemented as IOT App of Snap4City
  – be on other clouds and services
  – work on a large range of different protocols and kinds
  – have or not Snap4City libraries installed
• The **Internal IOT Orion Brokers** at Snap4City are used as gate for data ingestion and actuations. Since they are
  – connected with the IOT Directory and discovery of the Knowledge Base to make easy the production of Dashboards by wizard, Data Inspector;
  – Synchronized automatically with NIFI/OpenDistro(ElasticSearch) for the Automated Data Shadow and Indexing
  – Ready to be used by IOT App to subscribe for creating even driven IOT Apps, on IOT Edge and Cloud, etc...
  – Compatible and harmonized with FIWARE networks
• Direct Data Ingestion is also possible:
  – From data sources to Data Warehouse Storage of Snap4City, Snap4Industry

• Data Warehouse Storage includes: KB, and I&A, reported on right side can be acted via API REST Call
  – for direct feeding data into store and retrieval,
  – which can be exploited by:
    • IOT Applications
    • applications in Python, R Studio, Java
## Notation Terminology

<table>
<thead>
<tr>
<th>WHERE</th>
<th>Are synonymous at level of service which can be <strong>IOT device or entity</strong> with data</th>
<th>Are synonymous at level of the single <strong>attribute of the entity</strong>, device, service, etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOT Directory</td>
<td>IOT Device</td>
<td>Sensor, Actuator, Attributes, Values (value name)</td>
</tr>
<tr>
<td>Knowledge Base, ServiceMap, SmartCity API, ASCAPI</td>
<td>Service, ServiceURI, SURI</td>
<td>Attribute, Metric</td>
</tr>
<tr>
<td>DataInspector, Wizard, Dashboard</td>
<td>ValueName</td>
<td>Sensor, Sensor Actuator, ValueType</td>
</tr>
<tr>
<td>IOT Applications, Node-RED</td>
<td>ServiceURI, SURI</td>
<td>SURI and its real time results of the objects into the data structure</td>
</tr>
</tbody>
</table>

### How to access at SURI

**ServiceURI, SURI of a sensor device:**
- http://www.disit.org/km4city/resource/METRO759

**ServiceURI, SURI extended with attribute:**
- http://www.disit.org/km4city/resource/METRO759&metric=vehicleFlow
- In some cases
  - http://www.disit.org/km4city/resource/METRO759/vehicleFlow

See Part 3
IOT Broker Registration
(for External/Internal Orion Brokers or other Brokers)
Register IOT Broker
Snap4 technology is broker Agnostic. Most of the features are only accessible for FIWARE Orion Brokers

• IOT Brokers
  – You can test, view, and register, and also automatically deploy Orion Brokers
  – are associated with an Organization
    • Each Organization has a Knowledge Base of reference (KB, ServiceMap)
    • Each KB may host multiple Organizations and addresses multiple Geographic areas
  – can be compliant with
    • NGSI version: V1, V2-1, V2, etc...
      – with Snap4City Security or regular NGSI FiWare
    • other protocols as well such as: MQTT, COAP, AMQP, etc.
  – can
    • expose different authentication methods: K1/K2, Certificate, etc.
    • be accessible from IOT Devices and IOT App in Cloud only
    • be accessible from Internet to post data from outside, etc.
IOT Orion Broker Network: NGSI V1 and V2

• IOT Broker can be Internal (on Snap4City Cloud)
  – Registration of IOT Devices can be performed by the IOT Directory
  – Authentication is automatic, K1 and K2 are not needed since the security is performed via Access Token, M2M secure communication, on the basis of IOT App ownership
  – The NIFI Cluster automatically subscribes to all the entities on the Broker, to post data into the Data Shadow enriched with data of the KB

• IOT Broker can be External (managed by third party)
  – Registration of IOT Devices is managed by third parties
  – The registered IOT Devices can be collected and queried from the IOT Directory as well
  – The NIFI Cluster may automatically subscribes to all the entities on the Broker, to post data into the Data Shadow enriched with data of the KB

• IOT Brokers can be networked
  – Services, Service paths: for managing the IOT Broker network
  – Multi-tenant: more than one user/org on the same IOT Broker
- Orion Broker of V1 with **NGSI syntax of V1** + Secure Filter of Snap4city
- Orion Broker of V2 with **NGSI syntax of V2** + Secure Filter of Snap4city
- Orion Broker of V2 with **NGSI syntax of V1**
- Orion Broker of V2 with **NGSI syntax of V2**
IOT Devices / Brokers with Service / Service paths (SP)
IOT Orion Broker: Service and Service Paths

- The concept of Service Path (SP) allows to organize the IOT Devices managed by a Brokers in Directories and SubDirectories.
  - This implies that IOT Devices with identical ID may be located on different paths
- Path may be used for logica organization:
  - **Quadrant1 of the City**
    - Lamps: L1, L2, ........
    - Waste.............
  - **Quadrant2 of the city**
    - Lamps: L1, L2, ........
    - Waste.............
  - **Lamps**
    - Street 1. d1., d2..
    - street 2, ...
    - street 3, ..
  - **Waste**
    - street 1: d1, d2, ...
    - street 2
    - street 3
The Paths

- Have to start with /, only absolute
- Max 10 levels
- Max 50 chars for each level
- A Device can belong only to a unique level
Federation of Snap4City vs IOT ORION Broker

Federation of Snap4City vs IOT ORION Broker

Crate-DB

MongoDB

IoT Agent

Area 3

IoT Orion Broker

Quantum Leap

8668:8668

27017:27017

4200:4200

4041:4041

SSM2ORION

SUPER

Dashboard Builder

Snap4City Solutions

Hybrid Solutions

NGSI

1026:1026

FIWARE

Super

8668:8668

NGSI

4200:4200

27017:27017

8668:8668

Super

FIWARE

Super

Super

FIWARE

Super

FIWARE

Super

FIWARE

Super

FIWARE

Super

FIWARE

Super

FIWARE
IOT Devices / Brokers with Multi Tenant (MT)
IOT Orion Broker: MultiTenant

- The usage of Service can be used as Tenant of the same Broker
- The IOT Orion Broker and NGSI protocol do not allow to brose the whole set of Services, ServicePaths.
  - So that the single user may know its own Service which can be regarded as its own Tenant

Fiware-Service: tenant2
Fiware-ServicePath: /path1
Discovery of IOT Devices on External Brokers with SP/MT

- IOT Directory and Devices
  - My IOT Sensors and Actuators
  - IOT Sensors and Actuators
  - IOT Devices
  - IOT Devices Management
  - IOT Brokers
  - IOT Device Models
  - IOT Devices Bulk Registration
  - Ext. MS Broker Devices Discovery
  - Ext. Broker Devs Periodic Update
  - IOT Orion Broker Mapping Rules
External Orion Brokers

- External Orion Brokers can be Queryed to Discover the Service Paths (SP) and thus the Multitenant organization they have according to what is know by the IoT Directory.

- So that:
  - 1° step is to register the Orion Broken with information regarding SP
  - 2° to perform the IOT Device Discovery
    - This feature is accessible only for those kind of brokers to AreaManager
Register an External Multiservice Orion Broker
Device Discovery MT/SP on External Brokers
What is next

• Once the IoT Devices have been discovered,
  – the IoT Directory engine harvests Periodically the devices to understand their mapping on the known model and entities.

• Thus a number of miss-matches can be discovered and listed by device in the Ext. Broker Devs Periodic Update
Ext. Broker Devs Periodic Update

<table>
<thead>
<tr>
<th>Device Identifier</th>
<th>IoT Broker</th>
<th>Protocol</th>
<th>Format</th>
<th>Device Type</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>GroupA1DLC</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>SwitchingGroup</td>
<td>Invalid</td>
</tr>
<tr>
<td>KV-0121</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>StreetlightControlCabinet</td>
<td>Invalid</td>
</tr>
<tr>
<td>KV-0121-314</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>Streetlight</td>
<td>Invalid</td>
</tr>
<tr>
<td>KV-0121-315</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>Streetlight</td>
<td>Invalid</td>
</tr>
<tr>
<td>KV-0121-315_dailGearError</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>Alert</td>
<td>Invalid</td>
</tr>
<tr>
<td>KV-0121-315_dailGearError</td>
<td>onRCEAPELON-Tampere</td>
<td>ngs wMultiService</td>
<td>json</td>
<td>Alert</td>
<td>Invalid</td>
</tr>
</tbody>
</table>

Context broker: Antwerp
Model: custom
Edge-Gateway Type
Edge-Gateway URI

Suggest Modifications
Show active broker discovery
Retrieve devices
Solving the miss matches

- What the Miss Matches are, and how are shown?
  - See aside

- They can be solved in two manners:
  - Creating rules
  - Training the tools for each missmatch of each device-kind
Notes on Miss Matches

• MissMatches are practically all miss classification of ValueType and ValueUnit since the NGSI V2 does not have this information as mandatory, and thus messages never seen may not have it

• Once assigned a ValueType, ValueUnit for a ValueName of an IoT Device
  – All the messages of that Device will be managed automatically

• If you have thousands of devices it is possible to formalize a rule that can be applied to all Devices of MS thus shortening the training phase
IOT Device vs Time Series
What About IOT Devices, time series

A set of data coming from an IOT Device with multiple sensor become a time series of values for devices.

- For example: taking a new measure every 10 minutes (Red Lines)
- Non regular rates can be valid data as well.

Each new measure in Snap4City is conventionally time located in «dateObserved», which has to be Unique.

<table>
<thead>
<tr>
<th>dateObserved</th>
<th>Temp</th>
<th>Humidity</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-04-2020 10:30</td>
<td>34.5</td>
<td>23</td>
</tr>
<tr>
<td>02-04-2020 10:40</td>
<td>36.5</td>
<td>24</td>
</tr>
<tr>
<td>02-04-2020 10:50</td>
<td>36.0</td>
<td>22.5</td>
</tr>
</tbody>
</table>
IOT Device Model
IOT Device Data Model (1)

- **IOT Broker**
  - Name of the Brokers: among those registered
  - Protocol: NGSI, AMQP, MQTT, etc..
  - Format: CSV, JSON, XML.
  - Service/Tenant: ...........
  - ServicePath: ...........

- **Info**
  - Name (Identifier)
  - Model: Custom or Model ID
  - DeviceType: ..a string..
  - MAC address: ...optional...
  - Edge-GW: Raspberry, Android, ...
  - Edge-GW: URI
  - Producer
  - Owner
  - Freq: ..... Sec
  - Keys: K1, K2
## IOT Device Data Model (2)

### Add new device

<table>
<thead>
<tr>
<th>IOT Broker</th>
<th>Info</th>
<th>Position</th>
<th>Static Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Latitude</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Longitude</td>
<td></td>
</tr>
</tbody>
</table>

### Edit Model - ChargingStationModel

<table>
<thead>
<tr>
<th>General Info</th>
<th>IOT Broker</th>
<th>Static Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChargingStateValue</td>
<td>charging_state (Charging)</td>
<td>some coded status (state)</td>
<td>string</td>
</tr>
<tr>
<td>Refresh rate</td>
<td>900</td>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Static Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>stationStateValue</td>
<td>charging_station_state</td>
</tr>
<tr>
<td>Refresh rate</td>
<td>900</td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date/Time</th>
<th>values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date/Time observed</td>
<td>timestamp (Timestamp)</td>
</tr>
<tr>
<td>Refresh rate</td>
<td>900</td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
</tr>
</tbody>
</table>

Snap4City (C), January 2022
## IOT Device Model (3): Attributes

<table>
<thead>
<tr>
<th>Where</th>
<th>IOT Device Model</th>
<th>IOT Device</th>
<th>A Temporal Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOT Broker</td>
<td>Broker: <strong>OrionUNIFI</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOT Broker</td>
<td>Protocol: <strong>NGSI</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Info</td>
<td>ID: string</td>
<td>ID: “park45”</td>
<td>park45</td>
</tr>
<tr>
<td>Position</td>
<td>GPS: lat, long</td>
<td>GSP Position: 43.12, 11.34</td>
<td>GSP Position: 44.12, 11.12</td>
</tr>
<tr>
<td>Static attribute</td>
<td>Description: string</td>
<td>Description: “parking massaia”</td>
<td></td>
</tr>
<tr>
<td>Static attribute</td>
<td>Location: string</td>
<td>Location: “Via Massaia”</td>
<td></td>
</tr>
<tr>
<td>Static attribute</td>
<td>Civic Number: string</td>
<td>Civic Number: 3</td>
<td></td>
</tr>
<tr>
<td>Static attribute</td>
<td>MaxCapacity: number, cars</td>
<td>MaxCapacity: 456</td>
<td></td>
</tr>
<tr>
<td>Values</td>
<td>dateObserved: Timestamp</td>
<td></td>
<td>23-12-2019T20:13:12...</td>
</tr>
<tr>
<td>Values</td>
<td>FreeSlots: Integer, #</td>
<td></td>
<td>345</td>
</tr>
<tr>
<td>Values</td>
<td>Humidity: float, %</td>
<td></td>
<td>25.5</td>
</tr>
<tr>
<td>Values</td>
<td>Temperature: float, celsius</td>
<td></td>
<td>34</td>
</tr>
</tbody>
</table>
• **ID:** is the unique identifier for reconnecting Temporal Instances with registered IOT Devices

• **Static Attributes:**
  – Are typically associated with instances of the IOT Device. E.g.: You have a set of parking areas, each of them is located in a specific street, and has its one name, etc.
  – Different kinds of attributes can be set for each SubNature. Their definition has to be prepared into the Knowledge Base 🎈 for automated indexing.

• **Values:** they are time varying variables (temporal values/instances)
  – They change over time, the timestamp of the time series is conventionally «dateObserved» in Snap4City
  – In new *SensorMobile* HLT, also GPS can be changing over time as in the MyKPI

• **NOTE for:**
  – names/IDs: Spaces or strange characters are not allowed in the. Please use simple alfphanumeric strings, it is a limitation of many solutions including Orion Broker and increase interoperability of your data.
  – **Values of attributes and variables:** can be UTF8, but similarly, they do not accept: () <> “ ’ ; = into values
Using the IOT Device Model notes!!!

- Once performed the IOT Device Model, a number of IOT Devices can be produced using the model as a Template

  - **NOTE**: the produced IOT Devices are not going to change if the IOT Device Model is modified.

  - *Your biscuits are not changing if the template is modified after the printout*
IOT Device Management

(only for: RootAdmin and ToolAdmin)
IOT Device Management
IOT Device Registration
Activities for IOT Device Registration

• Registration of
  – N IoT Devices and their IoT Device Model via Data Table Loader
  – an IOT Device Manually from Zero or from an IOT Device Model
  – a set of IOT Devices with the same IOT Device Model from IOT App
  – a set of IOT Devices in BULK loading a CSV (with or without a reference IOT Model)

• The IOT Device registration implies the automated production of the Digital Twin Device into the Knowledge Base, which implies:
  – Activation of the Storage “DataShadow” for historical data access
  – Activation of all the relationships
  – Activation of Discovery mechanisms via IOT Directory, KB
  – Activation of Dashboard Wizard (after a few minutes), and Data Inspector
  – Etc.
Manual Registration of IOT Device or By Model

<table>
<thead>
<tr>
<th>Name</th>
<th>IO Broker</th>
<th>Protocol</th>
<th>Format</th>
<th>Device Type</th>
<th>Ownership</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARDUNO_ST_A201</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A202</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A203</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A205</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A206</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A207</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A209</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A210</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A211</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Search Device Location on Map

IOT Sensors and Actuators

<table>
<thead>
<tr>
<th>Name</th>
<th>IO Broker</th>
<th>Device Type</th>
<th>Value</th>
<th>Value Type</th>
<th>Health/Act Status</th>
<th>Last Read</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARDUNO_ST_A202</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A203</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A204</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A205</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A206</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A207</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A209</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A210</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARDUNO_ST_A211</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Manual Registration of IOT Device or By Model
IOT Discovery on IOT Application Development

Snap4City (C), January 2022
IOT Device Registration via IOT Device Model
Many IoT Devices?  IOT Device Model!!!

- **Prerequirements**: only for AreaManager users
- If you have a set of sensors with the same features,
  - you can create a model and then a set of instances (IoT Devices) in compliance with the model (not time consuming and avoiding errors)
- IoT Directory and Devices > IoT Device Models > ‘New Model’ button

Example: ChargingStationModel

Snap4City (C), January 2022
Add IOT/IOE Devices, exploiting an IOT Device Model
Add IOT/IOE Devices, exploiting an IOT Device Model

Just Buy an IOT Device and register: SigFOX, MQTT, NGSI (FiWare), ...
  – Attach them by
    • Models
  – A range of protocols, formats, approaches

Create your own devices:
  – Arduino,
  – Raspberry,
  – Android,
  – LoraWAN + Arduino,
  – etc.

Secure Communication: HTTPS, TLS (K1, K2), Certificates
Add IOT/IOE Devices, exploiting an IOT Device Model

Addition of Static Attributes of the IOT Device

Only if you enabled them into the IoT Device Model
IOT Device Registration via IOT Device Model in Bulk
From CSV → register IoT Devices in BULK

- Create a CSV from the CSV Model provided
- The columns must respect the CSV Model (every field present in the Model)
- One row of the CSV is one new IoT Device
- You have to create to create two keys (called k1, k2) that are necessary to read and write access to the device. They must be different each other.
- Each group of devices, that has the same IoT Model (data set), could/should have the same K1, K2. In this way, it is easier to read or write all the IOT Devices of the same set at the same time.
- These keys are in the UUID v4 format and can be generated online on this website: [https://www.uuidgenerator.net](https://www.uuidgenerator.net)

Available example: [https://www.snap4city.org/592](https://www.snap4city.org/592)
Register IoT Devices in BULK

- IoT Directory and Devices > IoT Devices Bulk Registration
- Select: Model, Broker
- Upload the CSV file
- Wait
- Verify the presence of your Devices in:
  - IoT Directory and Devices > IoT Devices

- [https://www.snap4city.org/289](https://www.snap4city.org/289)

Snap4City (C), January 2022
IOT Device Registration from IOT App (automation)
1) Model creation

Model name: Florence wifi average person
2) IoT Devices Creation

BLOCK: ‘IoTDirectory-new-device-from-model’
Model name: Florence wifi average person

3) Group Creation (more than 200 devices) -> put all the devices in the group and put them as ‘public’ (or they remain private)
4) Send RT data to the IoTDevices
5) Verify RT Data via Snap4City API or via ServiceMap
1) IoTModel

2) Static Flow to create IoTDevices

3) Add the license and Make Public the IoTDevices (according to the license)

3) Search for the Cameras on Map (ServiceMap)

5) Working on Dynamic Flow to save Average #people every 15 minutes for each IoTDevice

Snap4City (C), January 2022
Complete Examples
A Complete Example for Time Series: IOT Device Model + IOT Data Ingestion
I have created an IOT Device Model as:

Snap4City (C), January 2022
For Time Series
- **ValueName:** `dateObserved`
- **ValueType:** `timestamp`
- **ValueUnit:** `timestamp in millisecond`
- **DataType:** `string`
- E.g.: ISO string of the date-time
Please note for Time Series of IoT Devices

• Snap4City engine recognizes as time basis for the TimeSerie only 1 Variable with
  – ValueType as TimeStamp (in milliseconds)

• If you need more than one timestamp in milliseconds use:
  – ValueType as DateTime (in milliseconds)
From IOT Model I have created some instances: the IOT Devices

<table>
<thead>
<tr>
<th>Device Identifier</th>
<th>IOT Broker</th>
<th>Device Type</th>
<th>Model</th>
<th>Ownership</th>
<th>Status</th>
<th>Edit</th>
<th>Delete</th>
<th>Location</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>adminDev1</td>
<td>orionUNIFI</td>
<td>Ambiental</td>
<td>PUBLIC</td>
<td></td>
<td>active</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160543238906</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160543454047</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160631897023</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160629997473</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160773747380</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160759804347</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160775997465</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160777002089</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_160771247659</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Snap4City (C), January 2022
They have been created by «Add new Device»
IOT Device from IOT Model by Providing:

• NAME (it has to be unique)

• Select the IOT Model: «statuscorregione»
  – Thus the K1, K2 appears since the model is associated to an Orion Broker that needs to have them, the tool generate them for you but you can impose if you like
  – See in previous slide the ID name of the IOT Broker used

• Lat and Lon, GPS coordinates you can:
  – pick on the map
  – Write the coordinates manually and see the pin on map
Once Created, I may send a new data to it

- Get/See last message from Broker
- Generate a New Message towards the Device, Storage
- View IoT Device on map and its last value
Impose current date time on dateObserved

Send the Message to the Device, Storage
Once created the IOT Device you may send data on it

• You may create an IOT App, where:
  – Function: is preparing the JSON package
  – Block «Fi-Ware Orion OUT V1» or V2 is sending the data to the Orion Broker. Namely: «OrionUNIFI»
  – Please note that several version of IOT ORION Brokers and protocols exists:
    • So that you have to know which protocols you need to use for your broker
Settings?

• Certificates are automatically loaded at the first authentication
• Done!!

- IP if the Broker is in cloud (internal)
  - List of brokers is automatically provided
  - The K1, K2 is automatically provided if you are authenticated
- Symbolic address of IOT Broker can be taken from IOT Directory
NGSI versions

- Orion Broker of V1 with **NGSI syntax of V1** + Secure Filter of Snap4city

- Orion Broker of V2 with **NGSI syntax of V2**

- Orion Broker of V2 + Secure Filter of Snap4city

- Orion Broker of V1 with **NGSI syntax of V1**

- Orion Broker of V2 with **NGSI syntax of V2**
• A Json from the IOT App
  – NGSI V1
• ID: The Name of the IOT Device: «corveneto»
• Type as that define in the IOT Device when you created
• The Time stamp: “dateObserved” to have a time series data
  – “str” is a string with the date and time in standard ISO, such as,
  – “2020-08-04T04:00:00+02:00”
  – “2020-08-03T00:00:00.000Z”
• And the vector of “attributes”

```json
msg = { payload: {
  "id": "corveneto",
  "type": "misura",
  "attributes": [
    { "name": "dateObserved", "value": str, "type": "timestamp" },
    { "name": "stato", "value": "active", "type": "string" },
    { "name": "ricoverati_con_sintomi", "value": 12, "type": "integer" },
    { "name": "terapia_intensiva", "value": 34, "type": "integer" },
    { "name": "totale_ospedalizzati", "value": 34, "type": "integer" },
    { "name": "isolamento_domiciliare", "value": 334, "type": "integer" },
    { "name": "totale_attualmente_positivi", "value": 12, "type": "integer" },
    { "name": "nuovi_attualmente_positivi", "value": 33, "type": "integer" },
    { "name": "dimessi_guariti", "value": 22222, "type": "integer" },
    { "name": "deceduti", "value": 2, "type": "integer" },
    { "name": "totale_casi", "value": 2222, "type": "integer" },
    { "name": "tamponi", "value": 222222344, "type": "integer" }
  ]
}

return msg;
```
The differences are mainly on how the variable are provided:

```json
{
"id":"MyMobileDeviceTest",
"type":"misura",
"dateObserved":{"type":"timestamp","value":"2021-06-11T16:17:23.425Z"},
"status":{"type":"float","value":25}
"mydescription":{"type":"string","value":"see below the note for the forbidden characters"}
}
```

- **NOTE for:**
  - **names/IDs:** Spaces or strange characters are not allowed in the. Please use simple alphanumeric strings, it is a limitation of many solutions including Orion Broker and increase interoperability of your data.
  - **Values of attributes and variables:** can be UTF8, but similarly, they do not accept: () <= '<' ; = into values
• The **Timestamp**: “dateObserved” to have a time series data
  – “str” is a string with the date and time in standard ISO, such as,
  – “2020-08-04T04:00:00+02:00”,
  – “2020-08-03T00:00:00.000Z”

• In JavaScript you can obtain by using:
  – Var str = new Date().toISOString();
  – **Str** has to be the ISO date string of today-now (at the current time).
From date to ISOString with fuse aligned time

New oggi = new Date();
dateCET2Z(oggi).toISOString();

----

function dateCET2Z(date) {
  d = new Date(date).toLocaleString('nl-BE', {timeZone: 'Europe/Brussels'});
  offset = new Date(d).getTime() - new Date(date).getTime();
  return new Date(new Date(date).getTime() - offset);
}
Multi Series Widget coming from the same IOT Device

- Over on the serie label to highlight
- Click on the serie label to on/ok
- Over on the graph to see the values

Andamenti Nazionali e Regionali infezione COVID-19
Sulla base dei dati della protezione civile, elaborazioni DISITLab

per evidenziare gli andamenti di vostro interesse: eliminare le curve che non interessano selezionandole in legenda.
Alcuni dati in passato non sono pervenuti alla protezione civile

JSON for Authentication as well

```json
msg.auth = {
    "k1": "1ef0e5e8-yyyy-xxxx-9462-0aa4cfcf5e19",
    "k2": "b2b34425-yyyy-xxxx-818d-2d6cac2314a6",
    "apikey": "apikey",
    "basicAuth": "basicAuthKey"
};
```
You may use other functions from IOT Directory

- IOT Discovery in an area
- Query on IOT Directory
- Get IOT Device Info
- Registering an IOT Device from model
- Delegate an IOT Device
- Change Ownership of an IOT Device
Get IOT Device Info

- You can create smart IOT Applications that on the basis of the list of IOT Devices would request all what you need to load data into YOUR OWN IOT Devices including:
  - Service URI
  - K1, K2
  - Authentication
A Complete Example: Time Series, IOT Device Model, IOT Data Ingestion, Dynamic K1,K2 Management
Create/instantiate IOT Devices

A) New IOT Device: From Scratch or from IOT Dev Model

B) New IOT Device (simplified creation) from IOT Device Model
B) **IOT Device Model**

To exploit the simplified IOT Dev Creation, you have to create the IOT Device Model as here.
To exploit the simplified IOT Dev Creation, you have to create the IOT Device Model as here. Do once and create as many IOT Device you like in shorter steps.
IOT Devices

Search Device Location on Map

Broker Port: 8080
Visibility: MyOwnPublic
Format: json
MAC:
Producer: disit
Latitude: 43.76923

Payload
K2: 168473c-08bc-4a4e-a22b-aad5a4f08030

Snap4City (C), January 2022
Last messages from the brokers can be taken. They can be used on IOT App compliant with the NGSI V1 and V2 as msg.payload.
Examples of NGSI V1 and V2, JSON Payload

**NGSI V1**

```json
{
  "type": "misura", "id": "Psvg4",
  "attributes": [
    { "name": "dateObserved", "type": "timestamp", "value": "2020-11-06T13:25:12.191Z" },
    { "name": "latitude", "type": "float", "value": "43.76923" },
    { "name": "longitude", "type": "float", "value": "11.2659s" },
    { "name": "str1", "type": "string", "value": "la mia" },
    { "name": "val1", "type": "float", "value": "3.6" },
    { "name": "val2", "type": "float", "value": "5.77" },
    { "name": "val3", "type": "float", "value": "6.78" }
  ]
}
```

**NGSI V2**

```json
{
  "id": "Psvg4", "type": "misura",
  "dateObserved": { "type": "timestamp", "value": "2020-11-06T13:25:12.191Z" },
  "latitude": { "type": "float", "value": "43.76923" },
  "longitude": { "type": "float", "value": "11.26596" },
  "str1": { "type": "string", "value": "eccolo" },
  "val1": { "type": "float", "value": "4.3" },
  "val2": { "type": "float", "value": "4.6" },
  "val3": { "type": "float", "value": "" }
}
```
Opla! See them Created on ServiceMap
Saving Data in the IOT Device: Data Shadow

- Generation of random values of the 4 IOT devices
- Send the values to IOT Broker,
  - which in turn send automatically the value to the Storage
- Inspection of the IOT Device Information
Function Block

- Data values
  Random generation
- `dateObserved` with the actual datetime
- Example in NGSI V1
Send data to Broker

- This case is with K1 and K2 directly written into the Orion OUT V1 Node
msg.payload = "http://www.disit.org/km4city/resource/iot/orionUNIFI/DISIT/Psvg2b";return msg;
msg = { payload : { "devicename" : "Psvg2b" } }
return msg;
Exploiting IOT Directory for getting K1,K2

msg = { payload : {
    "id": flow.get("devid"),
    "type": "misura",
    "attributes":[
        { "name": "dateObserved", "value": data, "type": "time" },
        { "name": "val1", "value": rnd1, "type": "float" },
        { "name": "val2", "value": rnd2, "type": "float" },
        { "name": "val3", "value": rnd3, "type": "float" },
        { "name": "val4", "value": rnd4, "type": "float" },
        { "name": "str1", "value": "eccolo", "type": "string" }
    ],
    auth :{
        "k1": msg.payload.content.k1,
        "k2": msg.payload.content.k2
    }
}}

// payload in NGSI V1 format

// the values of K1, K2 arrive from
// the iotdirectory-get-device
The four IOT Devices on Dashboard

Custom Pins on Map - test GP

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=Mjk5MA==
A Complete Example: IOT Device Mobile
Moving Data can be collected by using:

- **MyKPI**: in which each MyKPI has a **ValueName**, Unit, Type, etc.. And also GPS location.

- **IOT Device in Mobility**: which generates a new HLT SensorMobile, creating a TimeSeries with changing value and GPS coordinates over time.
**MyKPI: Tracking of Devices and Mobiles**

- Real Time Trajectories for
  - Mobile Phone
  - Moving IOT Devices
  - OBU, Vehicular Kits
  - Multiple tracks
  - Day by day

- Micro Application
IOT Device in Mobility: which generates a new HLT SensorMobile, creating a TimeSeries with changing value and GPS coordinates over time.
IOT Devices in Mobility as time series

- A set of data coming from an IOT Device with multiple sensor become a time series of values for devices.
- For example taking a new measure every 10 minutes (Red Lines)
- Each new measure in Snap4City is conventionally time located in «dateObserved»
- LAT, LONG is changing over time

<table>
<thead>
<tr>
<th>dateObserved</th>
<th>Temp</th>
<th>Humid.</th>
<th>LAT</th>
<th>LON</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-04-2020 10:30</td>
<td>34.5</td>
<td>23</td>
<td>11.11</td>
<td>43.21</td>
</tr>
<tr>
<td>02-04-2020 10:40</td>
<td>36.5</td>
<td>24</td>
<td>11.13</td>
<td>43.12</td>
</tr>
<tr>
<td>02-04-2020 10:50</td>
<td>36.0</td>
<td>22.5</td>
<td>11.45</td>
<td>43.18</td>
</tr>
</tbody>
</table>

#### Message

- Humidity
- Temperature

Snap4City (C), January 2022
Sending data of Moving IOT Device

```javascript
msg = {
  payload: {
    "id": "MyMobileDeviceTest",
    "type": "misura",
    "dateObserved": {
      "type": "time",
      "value": data,
    },
    "latitude": {
      "type": "float",
      "value": lat,
    },
    "longitude": {
      "type": "float",
      "value": lon,
    },
    "status": {
      "type": "string",
      "value": "",
    },
    "var1": {
      "type": "float",
      "value": rnd2,
    },
    "var2": {
      "type": "float",
      "value": rnd3,
    }
  }
}
```
Real time device tracking

Moving and changing Dynamic PIN

IOT Devices, hardware-software integration
Snap4City Services also on IOT Edge!!!

- IOT Networks
- IOT Brokers
- IOT Gateways
- IOT Edge Devices
- IOT Devices

- IOT Applications
- Big Data Analytics, Artificial Intelligence
- Dashboards and Apps

Mainly fog computing and NGSI V1, V2 with security
Proprietary IOT Devices as well as Open Hardware / Open Software
**IOT Devices**

- LoraWAN + Arduino + I2C, NGSI
- Arduino, Wi-Fi, NGSI
- Snap4All
  - IOT Button
  - ESP, NGSI, Wi-Fi, BT
- Snap4All PAX Counter
- LoraWAN
- WIFI, NGSI, GPS

**IOT Edge Devices**

- IOT Edge
  - NodeRED: Raspberry Pi, NGSI, WiFi, RJ45,
  - NodeRED: Android, LINUX, Windows,
  - Gateway:
    - LoraWan
    - IOT Edge, NGSI, WIFI, RJ45, GPS

Any Sensor / Actuator Open to other protocols
IOT Dev Management: activities

- IOT Devices connected to Snap4 can be open or proprietary
- IOT Devices: a large range of protocols, formats and kind
  - IOT Devices (single or in bulk) are registered on IOT Directory and thus according to Knowledge base are registered to be used in IOT Applications, Dashboards, etc. with Shadow values, etc.
  - IOT Models are saved on IOT Directory for shortening the registration process
  - IOT Device healthiness is automatically monitored
- IOT Devices can be public or private
  - Full support of Proprietary protocols and devices
  - Providing Open Hardware and Open Software
- IOT Edge are devices with some computing capability, realized by using: Raspberry, Android, Linux, Windows, etc.
  - Release as: OS images on SD, APK for Android, Virtual Machine, Docker Container, etc.
- IOT Devices are connected via Secure Encrypted Mutual Authenticated channel of communication, TLS and certificates
IOT Device with Arduino

- Arduino Uno
- Wi-Fi shield, standard
- Mutual Authentication with certificates, or K1,K2,sha
- Secure encrypted connection, NGSI
- Open Source
- Fully Customizable
  - Any sensor
  - NGSI or any other protocol
Snap4All IOT Button

- Multi Wi-Fi
- Ready to use BLE
- ESP 32 based, cheap & easy
  - low/no energy consumption/ standby
- Mutual Authentication with certificates, or K1,K2,sha
- secure encrypted connection, NGSI
- Open Source, Fully Customizable
- HW extensible to sensors

version: 3

https://www.snap4city.org/276
https://www.snap4city.org/297 helpconfig

Snap4City (C), January 2022
• Can be connected (i) directly to Snap4City (data driven) or (ii) Indirectly via CNR IBE (only in push)
CNR IBE AirQino
• CNR developed a circuit board "AirQino", Arduino Shield compatible, integrated with low cost and high resolution sensors, dedicated to the monitoring of environmental parameters and air quality pollutants
  – Noise, Humidity, Temperature,
  – CO, CO₂, O₃, NO₂, CH₄,
  – road pavement quality (accelerometer) and the indices of well-being (globethermometer to calculate the index of thermal comfort) in an urban environment.

• The board integrates a microprocessor unit that acquires all the sensors installed and analyses fast data from accelerometer and noise sensor.

https://www.snap4city.org/658 how to set up on Snap4City

https://www.snap4city.org/download/video/tn/ARQuino-CNR.pdf
AirQINO vs SNAP4CITY

- Can be connected (i) directly to Snap4City (data driven) or (ii) Indirectly via CNR IBE (only in push)
Libelium Waspmote Plug&Sense
Smart Environment PRO
Libelium

- PM10
- Temp
- Humidity
- Pm2.5
- NO
- NO2
- CO2
- Etc.

https://www.snap4city.org/659 how to set up on Snap4City
Can be directly connected to Snap4City (data driven)
SigFox Integration
SigFOX: example of a development platform
• Proprietary Protocol, freq similar to Lora
• Final users, consumers may buy SigFox devices and subscribe to network to register their devices
• 1 msg per every 10 minute, max num msg per day, per year…
<table>
<thead>
<tr>
<th>Time</th>
<th>Delay (s)</th>
<th>Header</th>
<th>Data / Decoding</th>
<th>Location</th>
<th>Base station</th>
<th>RSSI (dBm)</th>
<th>SNR (dB)</th>
<th>Freq (MHz)</th>
<th>Frames</th>
<th>Callbacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-05-06 18:03:27</td>
<td>&lt; 1</td>
<td>0000</td>
<td>ack required</td>
<td>24</td>
<td>2BA8</td>
<td>-122.00</td>
<td>25.40</td>
<td>868.1491</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>296</td>
<td>-136.00</td>
<td>15.60</td>
<td>868.1420</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25F2</td>
<td>-119.00</td>
<td>32.05</td>
<td>868.1373</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td>2018-05-06 18:03:23</td>
<td>&lt; 1</td>
<td>0010</td>
<td></td>
<td>3/3</td>
<td>25F2</td>
<td>-120.00</td>
<td>15.57</td>
<td>868.1197</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2BA8</td>
<td>-122.00</td>
<td>29.05</td>
<td>868.1185</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2627</td>
<td>-141.00</td>
<td>10.40</td>
<td>868.1173</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td>2018-05-06 18:02:51</td>
<td>&lt; 1</td>
<td>0000</td>
<td>ack required</td>
<td>24</td>
<td>2BA8</td>
<td>-122.00</td>
<td>29.36</td>
<td>868.1357</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>296</td>
<td>-136.00</td>
<td>14.01</td>
<td>868.1347</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28B4</td>
<td>-134.00</td>
<td>17.38</td>
<td>868.1220</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td>2018-05-06 18:02:23</td>
<td>&lt; 1</td>
<td>0010</td>
<td></td>
<td>3/3</td>
<td>23DB</td>
<td>-110.00</td>
<td>11.80</td>
<td>866.1449</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>296</td>
<td>-137.00</td>
<td>14.80</td>
<td>866.1442</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28B9</td>
<td>-137.00</td>
<td>15.67</td>
<td>866.1447</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td>2018-05-06 18:01:48</td>
<td>&lt; 1</td>
<td>0000</td>
<td>ack required</td>
<td>24</td>
<td>23DB</td>
<td>-109.00</td>
<td>41.46</td>
<td>866.1553</td>
<td>1/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>209</td>
<td>-136.00</td>
<td>15.00</td>
<td>866.1550</td>
<td>3/3</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25C8</td>
<td>-139.00</td>
<td>11.81</td>
<td>866.1546</td>
<td>1/3</td>
<td>YES</td>
</tr>
</tbody>
</table>
• Possible connection in PUSH and PULL
• Ingestion via IOT Application on Cloud or on IOT App on Edge
• Suggested connection in PULL
• Can be connected Indirectly via SigFox gateway (in push or pull), here represented in PULL
LoRa

Lora IOT Gateway vs NGSI

Snap4City (C), January 2022
LoraWAN Dragino (Arduino)
Lora IOT Device, Arduino

- Arduino Uno, Mega
- LoraWan Connection
- Any sensor, + I2C
- Fully Customizable
- Open Source
- NGSI or any other protocols
- Gateway: Lora-NGSI Snap4

https://www.snap4city.org/216
Dragino a development platform for Lora
LoraWan Gateway/Edge out of the Box

• Raspberry Pi Based LoraWan Gateway
  – Powered 5V, Wi-Fi, RJ45, ...
  – GeoLocated GPS Antenna

• IOT Edge Snap4City
  – Including Node-RED, IOT APP

• Logical UpLink: LoraWAN TheThingsNetwork, NGSI V1, V2 (mutual authenticated Snap4City) toward IOT Broker

Physical UpLink as: Wi-Fi, RJ45
MQTT Integration
• Can be connected from/to MQTT devices or gateways in push
IOT Gateway / IOT Edge
Controlling Energy Power

Measuring Energy Consumption

IOT Edge: Node-RED + Snap4City

Measuring any kind of sensors values

Any kind of notification channel

Contextual (smart city/home) data, Data Analytics

Historical Data, Remote Control, Mobile App

Administrative Servers

Alexa: Voice Commands

DCS

Modbus

SCADA

OPC UA

PLC

ODBC

WiFi

Snap4City (C), January 2022
IOT Edge on Raspberry Pi

- Raspberry Pi
- Mutual Authentication with certificates
- Secure encrypted connection
- IOT Application inside
- Any sensor
- Any protocol from IOT devices
- NGSI or any other protocol
- Fully Customizable
- Local and Cloud Dashboard
- Special MicroServices

MicroServices:
- DHT
- ModBus
- any shield
- etc....
Raspberry for Edge
Snap4City on Raspberry Pi, IOT edge
IOT Edge Computing

City user
Would like to:
- Monitor and exploit temperature and humidity
- Manage sensors
- Perform edge computing
- Using these data for multiple applications

Steps:
1. Registering the device and sensors
2. Create flow on edge device using NodeRed with Snap4City, sending data to Broker
3. Use data from Broker on Snap4City IOT App
IOT Edge Snap4All App for Android

- **Android**, any version, App from: [https://www.snap4city.org/download/video/Snap4All.apk](https://www.snap4city.org/download/video/Snap4All.apk)
- **Mutual Authentication** with certificates
- **Secure encrypted connection**, NGSI
- **IOT Application** inside
- Any sensor + Local device sensors
- Any protocol from IOT devices
- NGSI or any other protocol
- **Fully Customizable**
- Local and Cloud Dashboard
- Special MicroServices

[https://www.snap4city.org/drupal/node/278](https://www.snap4city.org/drupal/node/278)
IOT Edge Snap4All App for Android

MicroServices:
- Snap4City
- Termux Snap4City specific
- etc.
Sii-Mobility: Dynamic Signage and Street Mng

Safe and resilient solution managing Degraded conditions

Control Room

Snap4City (C), January 2022
IOT for Mobility Infrastructure

• C. Badii, P. Bellini, A. Difino, P. Nesi, "Sii-Mobility: an IOT/IOE architecture to enhance smart city services of mobility and transportation", Sensors, MDPI, 2019

• https://www.mdpi.com/1424-8220/19/1/1/pdf
IOT Tracking Devices
PaxCounter devices

• **Fix PaxCounter LoraWan**
  – sniffing on: Wi-Fi, Bluetooth
  – Sending data via LoraWan

• **Mobile PaxCounter LoraWan**
  – sniffing on: Wi-Fi, Bluetooth
  – Sending data via LoraWan

• **Fix PaxCounter, multiple out**
  – Sending data via LoraWan and Wi-Fi
  – sniffing on: Wi-Fi, Bluetooth

https://www.snap4city.org/456
Programmable PAX counting

Mobile PAXCounter 01 in Antwerp

Begin 3:00
Finish 5:30

Cumulative Mode OFF

Cumulative Mode Active from 2019-09-23T00:00:00Z to 2019-09-23T19:59:59Z

Device in Cumulative Mode OFF
IOT App behind

Snap4City (C), January 2022
IOE – Vehicle Monitoring

CANBUS sniffer

Tuscany in a Snap Mobile App on Android
MyKPI: Tracking of Devices and Mobiles

- Real Time Trajectories for
  - Mobile Phone
  - Moving IOT Devices
  - OBU, Vehicular Kits
  - Multiple tracks
  - Day by day

- Micro Application

Snap4City (C), January 2022
Real time device tracking

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MzA1Ng==

Moving and changing Dynamic PIN
IOT Devices and IOT Edge (Self Training)

• A large range of Devices can be used on Snap4City:
  – Proprietary or Open HW/SW.
  – Devices of/for makers on which we provide Open source code

• Documentation and instructions:
  – TC9.4 - IOT application exploiting Edge computing with Raspberry
  – TC9.7 - Connection from LoraWan Dragino/arduino to Orion broker
  – Snap4City: Arduino & ESP8266 IOT Device NGSI
  – Snap4City IOT Devices Registration
  – Snap4All IOT Button: based on ESP32, NGSI compliant secure connection
  – IDE Setup for Snap4All IOT Button, and source code
  – Registering IOT Edge: example of Raspberry Pi, total security
  – Creating: IOT Device, Raspberry Pi based, totally compliant with Snap4City
Logging Internal Events
Event Logger

• There are API for Event Logger, REST API
  – They are automatically used by most of the Snap4City MicroServices
  – They log in standard Rsyslog API
    • https://www.snap4city.org/56

• The Logs regarding messages passed and usage are logged and accessed with the AMMA tools that is based on OpenDistro per ElasticSearch and Kibana.
  – Former version was made in Hbase and SOLR, and Banana

• Additional Logs events can be logged by using a dedicated MicroService in Node-RED, IOT Apps
AMMA

• Managing and Monitoring Data-Traffic in the BackOffice

• Data Traffic Analyzer
  – Business intelligence
  – Faceted searches
  – Drill down on time

• Several different views and details on data traffic among the main entities in the platform:
  – IOT APP
  – Storage
  – Data sources,
  – ….
Complexity in Smart City IOT Platforms

End to End security
– From IOT Devices to Dashboard (user interface)

- H2M
- M2M
The secure stack

Sensors, actuators

IoT Devices

Data Shadow

IoT Edge

IoT Firewall, IoT Context Broker

storage

Aggregator and distributor: On premise computing

Via MicroServices: Cloud computing

User Interface

Dashboard Builder

HTTPS

HTTPS

HTTPS

HTTPS

WSs

HTTPS

WSs

HTTPS

HTTPS

HTTPS

HTTPS

HTTPS
Intranet

Internet

WSs, HTTPS

IOT Devices

[**IOT Edge**]

IOT Broker

[IOT Application]

Dashboard Engine

WSs, HTTPS

WSs, HTTPS

WSs, HTTPS

Sensors/Actuators

Executing local computation

Making IOT data more available

Smart City Knowledge Base and RT data

Microservices Ext-Services

Executing permanent computation

User interface

On Browser

Cloud

Exe Services

Smart City IoT Platform Respecting GDPR Privacy and Security Aspects

CLAUDIO BADII, PIERRFRENCSCO BELLINI, ANGELO DIFINO, AND PAOLO NESI (Member, IEEE)
Department of Information Engineering, University of Florence, 50134 Florence, Italy
Corresponding author Paolo Nesi (paolo.nesi@unifi.it)

This work was supported in part by the European Union’s Horizon 2020 Research and Innovation Programme under Agreement 988996.

ABSTRACT The Internet of Things (IoT) paradigm enables computation and communication among tools that everyone uses daily. The vastness and heterogeneity of devices and their composition offer innovative services and scenarios that require a new challenging vision in interoperability, security, and data management. Many IoT frameworks and platforms claimed to have solved these issues, aggregating different sources of information, combining their data flows in new innovative services, providing security robustness with respect to vulnerability and respecting the GDPR (General Data Protection Regulation) of the European Commission. Due to the potentially very sensible nature of some of these data, privacy and security aspects have to be taken into account by design and by default. In addition, an end-to-end secure solution has to guarantee a secure environment at the final users for their personal data, in transit and storage, which have to remain under their full control. In this paper, the Snap4City architecture and its security solutions that also respect the GDPR are presented. The Snap4City solution addresses the full stack security, ranging from IoT Devices, IoT Edge on premises, IoT Applications on the cloud and on premises, Data Analytics, and Dashboarding, presenting a number of integrated security solutions that go beyond the state of the art, as shown in the platform comparison. The stress test also included the adoption of penetrations tests validating the robustness of the solution with respect to a large number of potential vulnerability aspects. The stress security assessments have been performed in a piloting period with more than 1200 registered users, thousands of processes per day, and more than 1.8 trillion of complex data ingested per day, in large cities such as Antwerp, Helsinki and the entire Tuscany region. Snap4City is a solution produced in response to a research challenge launched by the Select4Cities H2020 research and development project of the European Commission. Select4Cities identified a large number of requirements for modern Smart Cities that support IoT/IoE (Internet of Things/Everything) and the hands of public administrations and Living Labs, and selected a number of solutions. Consequently, at the end of the project after 3 years of work, Snap4City has been identified as the winning solution.

INDEX TERMS End-to-end, GDPR, IoT, security, smart city.

I. INTRODUCTION

IoT (Internet of Things) is becoming a disruptive technology, especially for city users of metropolitan areas. The vastness of IoT Devices, integrated in common objects, is becoming increasingly deeper. The addresses’ space for these devices would be enough to point any sensors of any devices at any moment without reservations. Different products that implement Low-Power Wide Area Networks (LPWANs) technologies for IoT introduced by SigFox and Sentriech (LoRa, Long Range) have been gaining interest and have been under intense deployment campaigns worldwide [1]. At the same time, short-range IoT devices based on technologies such as IEEE 802.15.4 or Bluetooth Low Energy (BLE, [2]) are sold in increasing quantities and are already able to support scenarios for smart homes, energy metering and industrial automation. On the other hand, the start of the diffusion of 5G devices and services is creating high expectations in networking IoT technologies, as the killer application of previous technologies in metropolitan areas.
Comparison With other platforms
## State of the Art Solutions vs Snap4City

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Security end-to-end, secure on IoT and Dashboards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open HW and Open SW</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Source end-to-end</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability IOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOT and Dashboards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Programming end-to-end</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multi Domain Semantic Platform</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard based Modules and IOT, Open Devices</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resource Sharing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Analytics integrated Dashboard</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dashboard 24/7, protected connection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limited</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MQTT, coap, http</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MQTT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limited</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MQTT, HTTP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:
- "N" indicates not supported.
- "Y" indicates supported.
- "(Y)" indicates partially supported.
- "Limited" indicates limited support.

---

**Snap4City (C), January 2022**
Unique of Snap4City Platform (1)

- Data ingestion and model
  - **Unified data model** (exploited in the Wizard and Knowledge base)
  - **Semantic Reasoner** modelling city entities, supporting semantic search, expert system, digital Twin, etc.
  - **IOT Directory** abstracting complexity of IOT Devices, Edge, Brokers
- Data Analytics and Data Processes
  - **Flexible and extensible IOT Applications**
  - **Data Analytic**: multiple programming languages,
- Visual Analytics, dashboarding, Apps
  - **Wizard**: expert system for immediate dashboard production matching data vs graphics representation
  - **Dashboards specialized** multidomain for Smart Cities
  - **Custom Widgets and Synoptics**
  - **Ready to use Mobile App, instant App, MicroApplication**
  - **Strategies** formalization supports
Unique of Snap4City Platform (2)

- Openness to any developers
  - Living Lab support for coworking, sharing, and delegating
  - Multitenacy, multiorganization and geoareas
  - Advanced Smart City APIs and MicroServices
  - 100% Open Source, Open hardware
- Security and Privacy
  - End-2-end encrypted communication, on devices, platform, ... dashboards
  - GDPR compliant privacy/security
- Non functional
  - on cloud and on premise, your private installation
  - Ready to use Appliance Virtual Machines and/or Containers for a modules and tools.
  - Flexible, Modular, Elastic, scalable and robust
THE FIWARE SMART CITIES REFERENCE ARCHITECTURE

SMART CITY GOVERNANCE SERVICES
- Complex Event Processing
- Big Data Algorithms
- AI Algorithms
- Advanced Data Mops
- Operation Dashboards

Processing Engines (Flink, Spark, Hadoop...)
- Wireless Mashup
- GIS
- Knowledge BI

Extended Data Publication Platform (e.g. CKAN)
- Open Data Portal
- Data Marketplace

OIDON CONTEXT BROKER
- STREET
  - Location
  - Traffic
  - Pollution
- PARKING SPACE
  - Location
  - No. Slots
- BUS
  - Location
  - Route
  - Next stop
  - Time next stop
- WASTE BIN
  - Location
  - Capacity
  - Level
  - Threshold
- SHARED CAR
  - Location
  - Status
  - Driver
- CITIZEN CLAIM
  - Location
  - Citizen ID
  - Description

IDS AGENT FRAMEWORK
- Alternative IoT Platforms (e.g. OneM2M)
- IoT Agent
- System Adapter
- System Adapter
- System Adapter
- System Adapter

DEPLOYED IoT NETWORKS, CAMERAS, ROBOTS
- Other Information Sources
- Vertical Solutions
- Keynotes

Snap4City (C), January 2022
• **Snap4City - Powered by FIWARE Solution & Platform:**
    - NGSI V1, V2 The IOT Orion Broker
    - IOT Orion Broker can connect JSON, MQTT, Lightweight M2M, LoraWAN, OPC, SigFOX, etc. see FiWare [https://www.fiware.org](https://www.fiware.org)
• **Snap4City - FIWARE Training Services:**
  - [https://marketplace.fiware.org/pages/solutions/03bccd83a0e1b0398ba7a0bf](https://marketplace.fiware.org/pages/solutions/03bccd83a0e1b0398ba7a0bf)
• **Snap4City - FIWARE Consultancy Services:**
• **Snap4City is compatible** with all the above protocols
  - via IOT Orion Broker,
  - via IOT Applications.
  - via direct connection on ETL processes on their corresponding IOT brokers, and/or
• **Snap4City is also compatible** with many other protocols, see the table reported in page: [https://www.snap4city.org/65](https://www.snap4city.org/65)
Overview


Snap4City: FIWARE powered smart app builder for sentient cities

Snap4City (C), January 2022
• In Snap4City you can chose to connect your devices at Snap4City Platform in different manners:
  – (a) directly to Snap4City with some Broker, or on IOT App, Brokers, MyKPI
  – (b) via an IOT Orion Broker (external IOT Broker or those provided by Snap4City), or
  – (c) via any third party IOT Brokers in any protocol you have.

• Snap4City has
  – Improved IOT Orion Broker with the so called Orion Broker Filter (Orion Broker Filter, NGSI Security Wrapper) which is a secure wrapper for NGSI V1 and V2 protocol for enforcing Mutual Authentication, Security, roles, etc.
  – Produced open hardware and open software NGSI Compliant: as
    • IOT Devices with mutual authentication and security based for NGSI on: Android, Arduino and ESP32, IOT Button, etc.
    • IOT Edge devices with mutual authentication and security based for NGSI on: Raspberry PI, Windows, Linux.
SNAP4CITY

URAL PLATFORM: SMART CITY IOT AS A SERVICE AND ON PREMISE

IOT APPLICATIONS - INSTANT APPS
- DATA DRIVEN APPLICATIONS
- REAL TIME PROCESSING
- BATCH PROCESSING
- ANY PROTOCOL & FORMAT

DASHBOARDS & APPLICATIONS
- CONTROL ROOM
- SITUATION ROOM
- OPERATOR DASHBOARDS
- BUSINESS INTELLIGENCE
- WHAT-IF ANALYSIS
- DECISION SUPPORT
- SIMULATIONS
- RISK ANALYSIS
- RESILIENCE ANALYSIS

MOBILE & WEB APPLICATIONS
- DEVELOPMENT KIT
- SUGGESTIONS
- MOBILE APPS
- MONITORING PANELS
- PLATFORM UTILITIES
- READY TO USE SMART APPLICATIONS

MICROSERVICES & ADVANCED SMART CITY API
- IOT DIRECTORY
- SERVICE MAP
- RESOURCE MANAGER
- DATA GATEWAY
- IOT PLATFORM
- ETL

LIVING LAB - DEV TOOLS - COWORKING
- TEST CASES
- SCENARIOS
- VIDEOS
- HACKATHONS
- OPEN SOURCES
- COMMUNITY OF CITIES
- TRAINING TUTORIALS
- COMMUNITY MANAGEMENT

BIG DATA - DATA ANALYTICS
- PREDICTIONS
- ANOMALY DETECTION
- WHAT-IF ANALYSIS
- TRAFFIC FLOW RECONSTRUCTION
- ORIGIN-DESTINATION MATRICES
- SOCIAL MEDIA ANALYSIS
- OFFER VS DEMAND ANALYSIS
- ENVIRONMENTAL DATA ANALYSIS AND PREDICTIONS
- REAL TIME HEATMAPS
- ROUTING
- ALERTING
- EARLY WARNING
- PERSONAL AND VIRTUAL ASSISTANTS
- SMART SOLUTIONS
- SMART SHARING
- PARTICIPATORY

DATA ANALYTICS TOOLS - MICRO-APPLICATIONS
- KM4CITY DATA AGGREGAT KNOWLEDGE BASE
- EXPERT SYSTEM OF THE CITY
- BIG DATA STORE

IOT MNG - DATA MNG - DATA INSPECTOR - PROCESS MNG - USER ENGAGEMENT - GDPR MNG ...

OHS CITY UTILITIES OPEN DATA LEGACY & EXTERNAL SERVICES PERSONAL DATA IOT / IEO BROKERS KPI INDUSTRY 4.0 SOCIAL MEDIA
Dashboards and Apps

IOT Apps

Data Analytics, Artificial Intelligence

IOT Brokers

Node-RED

WorkFlow

KPI, POI, MyKPI, ...

API, External Services

Web Scraping

BIM

Big Data

KB

LD, LOD

GIS

FIWARE

ckan

IOT Brokers

IOT Broker

IOT Broker
Snap4City/Industry Detailed Architecture

- Orion Context Broker
- GW-NGSI
- Data Storage
- Data Analytic
- Predicting
- Planning
- IoT App
- IoT Devices
- Orders
- Energy Services
- Transportation
- External Services
- IoT Devices from the field
- DCS/SCADA
- Production Parameters
- Dashboards, Visual Analytics, Synoptics, 3D, Maps
- KeyCloak, LDAP
- Business Logic
- DCS Real Time · Setpoint
- CKAN
- GIS and ResM
- BPM & BIM
- IoT Directory
- Snap4City Dashboard Builder
- BPM & BIM
- GIS and ResM
- CKAN
- IoT Directory
- Snap4City Dashboard Builder
- KeyCloak, LDAP
- Business Logic
- DCS Real Time · Setpoint
- CKAN
- GIS and ResM
- BPM & BIM
- IoT Directory
- Snap4City Dashboard Builder
## Functional: FIWARE ref arc wrt Snap4City solutions

<table>
<thead>
<tr>
<th>FIWARE ref arc smart city</th>
<th>Snap4City</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple Protocols: IoT, Databases, etc..</td>
<td>10 on IOT, Limited on databases, etc.</td>
</tr>
<tr>
<td>Large set of high level types: maps, trends, heatmaps, traffic, trajectories, scenarios,...</td>
<td>No</td>
</tr>
<tr>
<td>Integration with workflows, BPM</td>
<td>Not Supported</td>
</tr>
<tr>
<td>Integration and Modeling Digital Twin BIM</td>
<td>Not Supported</td>
</tr>
<tr>
<td>Integration with GIS: WFS, WMS</td>
<td>Not fully supported</td>
</tr>
<tr>
<td>Integration with Heatmaps and Satellite</td>
<td>Partially, not calibrated</td>
</tr>
<tr>
<td>Integration with Satellite</td>
<td>not supported</td>
</tr>
<tr>
<td>Smart City API</td>
<td>no</td>
</tr>
<tr>
<td>Open Data Management</td>
<td>Partial with CKAN</td>
</tr>
<tr>
<td>Federation of platforms</td>
<td>Partial on brokers</td>
</tr>
<tr>
<td>Semantic model and queries</td>
<td>No, probably with NGSI-LD in the future</td>
</tr>
<tr>
<td>Multiple kinds of IoT Brokers</td>
<td>No, only agents</td>
</tr>
<tr>
<td>FIWARE ref arc smart city</td>
<td>Snap4City</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data Transformation</td>
<td>Coding</td>
</tr>
<tr>
<td>Data Analytics</td>
<td>No</td>
</tr>
<tr>
<td>on line development</td>
<td>No, limited</td>
</tr>
<tr>
<td>Dashboard on data</td>
<td>Grafana no LDAP</td>
</tr>
<tr>
<td>Dashboard Widgets</td>
<td>Limited, no custom, coding needed</td>
</tr>
<tr>
<td>Real Time end-to-end from Dashboards to any other channel, event driven</td>
<td>No, very limited</td>
</tr>
<tr>
<td>Multi Data Map</td>
<td>Limited with non OS</td>
</tr>
<tr>
<td>MicroApplications</td>
<td>No</td>
</tr>
<tr>
<td>Auditing, Assessment, accounting</td>
<td>No, no, no</td>
</tr>
<tr>
<td>Multitenacy on data management</td>
<td>No only on broker</td>
</tr>
<tr>
<td>Living Lab for creating/managing communities/groups</td>
<td>Not supported</td>
</tr>
<tr>
<td>Report generation/management</td>
<td>No</td>
</tr>
</tbody>
</table>
- **Open Data:**
  - Data gate, federation of Open Data Portals
  - IOT App, ETL proc(PULL)

- **IOT Networks:**
  - IOT Application processes, data driven or PULL
  - IOT Brokers (Push) → IOT Shadow

- **Web Pages:**
  - Web scraping, crawling processes

- **Satellite data**

- **Social media:** Twitter, Facebook, ...
  - Twitter Vigilance, IOT App

- **Mobile Apps**
  - Smart City API

- **Files upload:** CSV, Excel, etc.
  - IOT Applications, ETL

- **REST API, WS, FTP, LD, LOD, etc.**
  - IOT Applications, ETL

- **Data base accesses**
  - GIS: WFS, WMS
  - ETL, IOT Application

---

**Any kind of data and flows**

![Diagram showing data flows and various data sources and services including IoT Devices, IOT Brokers, Web Scraping, DataGate, IOT Apps, ETL, REST API, WS, FTP, LD, LOD, External Data Stores, and GIS data, Maps, etc.](image-url)

**Snap4City (C), January 2022**
Snap4City and FiWare integration

• A) IoT Orion Broker as an External Broker of a Snap4City platform
  – Devices are mainly managed by Orion Broker only
  – IoT Directory can harvest devices on Broker to registered them

• B) IoT Orion Broker is an Internal Broker of a Snap4City platform
  – This implies that Snap4City facilities are exploited for:
    • IoT Devices registration, IoT discovery, Ontology, Bulk registration, optimization of stored data, adaptation, filtering control, etc.
    • All the devices are registered into IoT Directory that performs the registration on both IoT Orion Broker and KB automatically

• C) Federation of an IoT Orion Broker with storage by using SSM2ORION
  – Devices are managed by Orion Broker only

• D) hybrid solutions in which Web and Mobile App can exploit both Orion API and Snap4City services and API
Snap4City IoT Registration and Access

A range of IoT Brokers and protocols

A range of other data sources

IoT Agent
IoT Orion Broker
MongoDB

Data Connectors

IoT Directory
ServiceMap
Smart City API

Authentication and Authorization

Dashboard Builder

Mobile Apps

Dashboards
A Mobile App may refer to one Smart City API Server (for Area 1) via SUPER and receive data from the Federated SUPERS (Area 2) if navigation, queries, etc. are leading to discover out of the addressed KB.

- SUPER can be used for creating redundant and/or balanced distributed solutions for Federated KB. See Area 2, the two KB in the front.
- Federated SUPER ServiceMap can have overlapped KB even totally.
- A Mobile App can be developed to support multiple Smart City API servers, for balancing and

The usage of Super (ServiceMap) is not mandatory so that separate services can be produced as well.

- SuperServiceMap and ServiceMap presents the same Smart City APIs.
Area 3
Federation of Snap4City vs IOT ORION Broker

IoT Agent

Federation

IoT Orion Broker

Crate-DB

MongoDB

Quantum Leap

8668:8668

27017:27017

4200:4200

8668:8668

4041:4041

1026:1026

SSM2ORION

Super

Dashboard Builder

NGSI

Hybrid Solutions

Snap4City Solutions
# Non Functional

## FiWare OS Solutions wrt Snap4City solutions

<table>
<thead>
<tr>
<th>FiWare</th>
<th>Snap4City</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security</strong></td>
<td>Yes: End to end, TSL and dashboards, event driven, mutual authentication, Access Token, OpenID Connect</td>
</tr>
<tr>
<td><strong>Privacy</strong></td>
<td>Yes: GDPR compliant full stack</td>
</tr>
<tr>
<td><strong>Access Control, authorization</strong></td>
<td>Yes: User Roles, and management tools</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Full stack Open Source</strong></td>
<td>Yes: open source also application level</td>
</tr>
<tr>
<td><strong>Full Modular</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Interoperable</strong></td>
<td>Yes at all levels, in all modules, 100% open source</td>
</tr>
<tr>
<td><strong>Full training course</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Examples and code shared</strong></td>
<td>Yes</td>
</tr>
</tbody>
</table>
Two Main Lines for Dashboarding are present

- **Dashboard Builder of Snap4City**
  - For accessing and browsing data on: OpenDistro (ElasticSearch), Mongo, MySQL, Smart City API, Super and thus from federated Smart City API, etc.
  - Supports sensors/actuators: data driven data, maps in extended manner, data driven widgets, large collection of widgets, direct IoT Connections, custom widgets, animated PIN on maps, a large set of panel/widgets, etc.
  - Very simple to be used for control room, decision makers, situation rooms, operators, etc.
  - Very well integrated with IoT App, Custom widgets, animation, external services.
  - Very simple to be customized for non programmers since all the tools are visual.
  - Support for GDPR and deep control of access.
  - Can integrate Kibana/Grafana Views into a Widget

- **Kibana** (so called DevDash, AMMA and recently My Dashboard (Dev))
  - Kibana, also accessible as Grafana
  - For accessing and browsing data on OpenDistro (ElasticSearch) storage and other sources supported
  - No Support for real time event driven widgets/panels, actuators and synoptics, no sophisticated maps, etc.
  - Not simple for control room, decision makers, etc.
  - Not integrated with IoT App, Custom widgets, animation, external services.
  - Oriented to developers, complex production of custom views, etc.
  - Partial support of GDPR and deep control of access.
# Snap4City Dashboard Builder vs Kibana

## Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Snap4City Dashboard Builder</th>
<th>Kibana, Grafana</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large Collection of Widgets</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Custom Widgets SVG of any kind, full defined process for customization</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Real time event driven widgets and data</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Business Logic for data transformation with visual programming: Node-RED</td>
<td>YES</td>
<td>Some coding</td>
</tr>
<tr>
<td>Maps with custom PIN, bubbles, animated and moving, etc.</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Maps with paths, shapes, traffic flow, scenarios, routing, heatmaps, what-if, ...</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Maps with Orthomaps from WFS, WMS, GIS connection, etc.</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>TV camera integration and selection</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Widgets for business logic integration on real time: buttons, selector, switch, etc.</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Kiviat, Spider net, Calendar</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Typical Time Trends: day hours, month week, month days, .....</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Time Trend Compare: day, eek, month, year</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Selectors/Menus: text, icons, etc., also in connection with IOT APP, Node-RED</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Full control of graphic layout, font, colours, refresh per widget, etc.</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Iframe integration of third party widgets and web pages, nesting dashboards, embedding Kibana</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Connection among multiple Dashboards and Widgets</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Synchronization with Video Wall, and Operators Views</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Multiseries, bar lines, charts, pie, donut, simple selectors, trends, etc., also from business logic</td>
<td>YES</td>
<td>Limited</td>
</tr>
<tr>
<td>Single content, string, html, any data, etc.</td>
<td>YES</td>
<td>Limited</td>
</tr>
<tr>
<td>Special widgets: Weather forecast, civil protection, road plates, Twitter, etc...</td>
<td>YES</td>
<td>Nothing</td>
</tr>
<tr>
<td>Faceted search</td>
<td>possible with selectors</td>
<td>YES</td>
</tr>
</tbody>
</table>
Acknowledgements
2020
- Smart Tourism
- 6 Pilots
- Data Analytics
- Extended platform

2021
- PC4City (2020-21) Monitoring Terrain
- Smart Ambulance (2021-22)
- Enterprise (2021-22) Industry 4.0
- Almafluida Industry 4.0 (2021-22)
- AMPERE (2021-22) Industry 4.0

2022
- SYN-RG-AI SmartCity
- PRETTO Industry 4.0
- Smart Mobility
- PISA, PUMS
- Living lab
- Smart Light
- Sweden
Main Organizations/areas

- Antwerp area (Be)
- Capelon (Sweden: Västerås, Eskilstuna, Karlstad)
- DISIT demo (multiple)
- Dubrovnik, Croatia
- Firenze area (I)
- Garda Lake area (I)
- Helsinki area (Fin)
- Livorno area (I)
- Lonato del Garda (I)
- Modena (I)
- Mostar, Bosnia-Herzegovina
- Pisa area (I)
- Pont du Gard, Occitanie (Fr)
- Roma (I)
- Santiago de Compostela (S)
- Sardegna Region (I)
- SmartBed (multiple)
- Toscana Region (I), SM
- Valencia (S)
- Venezia area (I)
- WestGreece area (Gr)
Main running instances (11/21)

- Sii-Mobility → mobility and transport, sustainability
- REPLICATE → ICT, smart City Control room, Energy, IOT
- RESOLUTE → Resilience, ICT, Big Data
- GHOST → Strategies, smart city
- TRAFAIR → Environment & transport
- MOSAIC → mobility and transport
- WEEE Life → Smart waste, environment
- Smart Garda Lake → Castelnuovo del Garda, SMARTEA
- 5G → Industry 4.0 vs SmartCity
- Green Impact → Industry 4.0, Chemical Plant, control and plan
- SmartBed (Laid) → smart health
- Green Field Peas (Soda) → Industry 4.0, Chemical plant
- MobiMart and PISA Agreement → data aggregation, mobility and transport, Living Lab
- Lonato del Garda → smart parking, environment
- Herit Data → tourism, culture and management
- ISPRA JRC → site management and services
- Capelon (Sweden) → smart light solutions
- PC4City → land slide monitoring and predictions
- Italmatic → industry 4.0 production control
Acknowledgements

- Thanks to the European Commission for founding. All slides reporting logo of Snap4City https://www.snap4city.org of Select4Cities H2020 are representing tools and research founded by European Commission for the Select4Cities project. Select4Cities has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation Programme (grant agreement n° 688196).

- TRAFAIR is a CEF project. All slides reporting logo of TRAFAIR project are representing tools and research founded by the EC on CEF programme http://trafair.eu/

- Thanks to the European Commission for founding. All slides reporting logo of REPLICATE H2020 are representing tools and research founded by European Commission for the REPLICATE project. REPLICATE has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation Programme (grant agreement n° 691735).

- Thanks to the European Commission for founding. All slides reporting logo of RESOLUTE H2020 are representing tools and research founded by European Commission for the RESOLUTE project. RESOLUTE has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation Programme (grant agreement n° 653460).

- Thanks to the MIUR for co-founding and to the University of Florence and companies involved. All slides reporting logo of Sii-Mobility are representing tools and research founded by MIUR for the Sii-Mobility SCN MIUR project.

- Km4City is an open technology and research line of DISIT Lab exploited by a number of projects. Some of the innovative solutions and research issues developed into projects are also compliant and contributing to the Km4City approach and thus are released as open sources and are interoperable, scalable, modular, standard compliant, etc.
On Line Training Material (free of charge)

<table>
<thead>
<tr>
<th>PDF</th>
<th>Interactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video1</td>
<td>Video2</td>
</tr>
<tr>
<td>duration</td>
<td>2:55</td>
</tr>
<tr>
<td>4th part (*)</td>
<td>Data Analytics</td>
</tr>
<tr>
<td>5th part (*)</td>
<td>System and Deploy Install</td>
</tr>
<tr>
<td>6th part (*)</td>
<td>Smart City API: Web &amp; Mob. App</td>
</tr>
</tbody>
</table>

[https://www.snap4city.org/577](https://www.snap4city.org/577)
Be smart in a SNAP!

www.snap4city.org

CONTACT
DISIT Lab, DINFO: Department of Information Engineering
Università degli Studi di Firenze - School of Engineering
Via S. Marta, 3 - 50139 Firenze, ITALY
https://www.disit.org

Email: snap4city@disit.org
Office: +39-055-2758-515 / 517
Cell: +39-335-566-86-74
Fax: +39-055-2758570