BE SMART IN A SNAP!

Smart IOT Applications & IOT Networks

October 2020, 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

October 2020, Course
https://www.snap4city.org/577

Paolo Nesi, paolo.nesi@unifi.it
https://www.Km4City.org
https://www.disit.org
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)
Free Trial

• Register on **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>Inter active</td>
<td>Video1</td>
<td>Video2</td>
<td>Video3</td>
<td>Video4</td>
<td>Video1</td>
<td>Video2</td>
<td>Video3</td>
</tr>
<tr>
<td></td>
<td>YouTube</td>
<td>YouTube</td>
<td>YouTube</td>
<td>YouTube</td>
<td>YouTube</td>
<td>YouTube</td>
<td>YouTube</td>
</tr>
<tr>
<td></td>
<td>2:55</td>
<td>3:16</td>
<td>3:41</td>
<td>none</td>
<td>2:00</td>
<td>2:48</td>
<td>none</td>
</tr>
<tr>
<td></td>
<td>duration</td>
<td>duration</td>
<td>duration</td>
<td>duration</td>
<td>duration</td>
<td>duration</td>
<td>duration</td>
</tr>
</tbody>
</table>
General Overview of the full Course

• **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, etc.
• **6th part:** Snap4City Architecture, How To Install Snap4City
• **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 Solution**
- Managing IOT Applications, Dashboards, Solution, Snap4Home, Snap4Industry, Industry 4.0, etc.

**Creating IOT Applications**
- Creating and demo
- IOT App = Node-RED + Snap4City solution, demo, Synoptics
- IOT App vs Data Analytics: R-Studio, Python
- IOT App self training, IOT App examples, web scraping,
- IOT App vs Synoptics in real time data driven

**From Simple to Data Processing IOT Applications**
- Demo + exercises, Data processing with IOT Applications
- Integration/Automating with Ticketing Systems Workflow
- Integration/Automating with Twitter Vigilance for social media analysis
- Integrating/automating with CKAN open data portals
- Automated production of MicroServices for IOT app from External REST CALL APIs

**IOT Network Support**
- Aim of IOT Networks, IOT Directory, IOT Broker Registration,
- IOT Device Model, IOT Device Registration, A complete example

**Comparison with other Platforms and Fi-Ware**
- FiWare vs Snap4City
- Snap4City Self training sources

**IOT Devices**
- Open and Proprietary devices, Open HW and Open SW
- IOT Devices, IOT Gateway, IOT Edge
- IOT Tracking devices

**IOT end-2-end Secure Stack**

**Acknowledgments**
Living Lab Accelerating

GO!

Community Building

City Operators

agreements

networking

tutorials

documentation

Inhouse companies

Tech providers

workshops

Category Associations

Corporations

Advertisers

Community

subscription to applications

collaborations

personal services

City Users

Manage Apps & Dashboards, User Engagement

Promote Applications & Dashboards

Monitor City Platform

Connect IOT/IOE

Connect external Services

Data Ingestion and Analytic algorithms

Advanced Smart City API, MicroServices

Produce City IOT Applications & Dashboards

Produce Apps and Dashboards for City Users

Help desk

Case Studies

Research groups

Partnerships

events

Large Industries

Early Adopters

Start-ups

Growth

Licensing, Gold services
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
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 App for Smartening Solutions
SNAP4CITY

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

LIVING LAB - DEV TOOLS - COWORKING
- IoT Directory
- Service Map
- Resource Manager
- Data Gate
- R Studio
- ETL

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 AGGREGATE KNOWLEDGE BASE - EXPERT SYSTEM OF THE CITY - BIG DATA STORE

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

OIS | CITY UTILITIES | OPEN DATA | LEGACY & EXTERNAL SERVICES | PERSONAL DATA | IOT / IOE | BROKERS | KPI | INDUSTRY 4.0 | SOCIAL MEDIA
Snap4City: Builder of Sentient Cities Solutions

Dashboards with data driven IOT Applications enforcing intelligence

IOT and data World

IOT Applications

Dashboards and Apps

My IOT Devices

Big Data Analytics, Artificial Intelligence
Snap4City Services also on IOT Edge!!

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

IOT Applications
- IOT Edge Devices
  - Big Data Analytics, Artificial Intelligence
- IOT Brokers
  - Mainly fog computing and NGSI V1, V2 with security

Dashboards and Apps

Snap4City (C), October 2020
Snap4City Architecture vs 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

Back-End
- Knowledge base
  - Semantic reasoners
- Indexing and aggregating
  - 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

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

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

Back-End
- Knowledge base
  - Semantic reasoners
- Indexing
  - Elastic search
- End aggregating

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, Widgets, MicroApps
- User interface, Interactive Dashboard
- Drill down, heatmaps

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


https://www.snap4city.org/65
IOT Data Driven

Snap4City IOT Brokers

Managing Public and Private IOT/IOE Devices

Towards any IOT Device and/or Dashboard

Real Time

Snap4city Platform storage for «Data Shadow» and much more

Real Time + Historical

Dashboards also provide rendering for sensor values
IOT App Smartening Dashboards
**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)
Estimation of the mean waiting time at bus stops
Antwerp
IOT App Smartening Devices and Dashboards
PaxCounter devices

- Fixed PaxCounter LoraWan
  - Based on Wi-Fi- Bluetooth
- Mobile PaxCounter LoraWan
  - Based on Wi-Fi- Bluetooth
- Fixed PaxCounter (LoraWan+Wifi out)
  - Based 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), October 2020
IOT App Smart Parking
IOT App Smart Industry 4.0
Snap4Industry
DISIT Lab, Distributed Data Intelligence and Technologies
Distributed Systems and Internet Technologies
Department of Information Engineering (DINFO)
http://www.disit.dinfo.unifi.it
http://www.disit.org

DCS
SCADA
PLC
PLC/RTU

Admin
IoT Broker
IoT Devices/Edge
Industry Plant1

Fleet management

IoT Broker
IoT Devices/Edge

IoT Broker
IoT Devices/Edge

SECURE

Internet

Dashboards and Apps
IOT Applications
Big Data Analytics, Artificial Intelligence

Control and Supervision on Multiple Supply Chains
Industry 4.0 as a Service

External Services
Snap4City Services also on IOT Edge!!!

IOT Networks

IOT Gateways

IOT Brokers

IOT Edge Devices

IOT Applications

Dashboards and Apps

Big Data Analytics, Artificial Intelligence

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

Smart City data from many sources

IOT Applications

IOT Data Shadow Snap4City

Big Data Analytics, Artificial Intelligence

ModBus to Snap4City Gateway Edge

5G network devices

Dashboards and Apps

IOT Data

5G network devices

Smart City data from many sources

ModBus to Snap4City Gateway Edge

5G network devices

Dashboards and Apps
Controlling Energy Power

Measuring Energy Consumption

IOT Edge: Node-RED + Snap4City

Measuring any kind of sensors values

WiFi

Any kind of notification channel

DCS

Administrative Servers

Alexa: Voice Commands

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

Snap4City (C), October 2020
IOT App Smart Industry 4.0
ModBus Integration
Devices

- A large range of devices: sensors and actuators
- Over serial and IP

Temp, Humidity

USB -- ModBus

8 Relais, actuators

Snap4City (C), October 2020
IOT App vs Smart Home
Snap4Home
Prato

Smart City vs Smart Home Estra
Snap4Home

Sonoff: Controlling Energy Power

Philips Hue: Controlling Lights

Hue: Motion Control / Alarm

Measuring Energy Consumption

TP Link: Controlling / Measuring Energy Plugs

Alexa: Voice Control

IOT Edge: Raspberry pi: Node-RED + Snap4City

Controlling Motors

Measuring Temperature and Humidity

Garage Door

Window Roller Shutters

Controlling Irrigators

Alarm sound and light

Environmental Contextual data from the city

Historical Data, Remote Control, Mobile App

My house

Living Room1 Room2 Room3 Garden Alexa Garage Windows

Energy Consumption

Snap4City (C), October 2020
IOT Edge:
Raspberry pi: Node-RED + Snap4City MicroService Library

Environmental Contextual data from the city. Historical Data, Remote Control, Mobile App
Snap4Home

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, Humidity, light in the room

Monitoring: CPU clock, status

5G gateway

Internet

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

Snap4City (C), October 2020
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

Sab 13 Giu
13°C / 23°C

Dom 14 Giu
Temp N/A

Lun 15 Giu
Temp N/A

Andamento di NO2

Privacy Policy  Cookies Policy  Terms and Conditions  Contact us

Snap4City (C), October 2020
IOT App vs Smart City Solutions
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

5.1
IOT Application with City Dashboard
simple development

Snap4City (C), October 2020
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

On road operator app
Events reporting

+ Risultati

Taxi Park
→ Taxi park

Antwerp
Helsinki
Florence
Current
Position

Blue
Green
Yellow
Red

PeopleNumber

GreenCode
YellowCode
RedCode

msg.payload

serviceSearch

setFlowLastPosition

setFlowPeopleNumber

generateOperatorEvent

SendOperatorEvents

WebApp

connected to ws://dashboard.snap4city.com
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

1648 nuovi, att. positivi
75528 Attualmente, positivi
14661 deceduti
19758 (Attuali guariti)

Andamento Contagio Coronavirus in Italia

variazione percentuale giornaliera Decessi (in alcune Regioni)

variazione percentuale giornaliera dei Casi (Nazionali)

Totale, Attuali Casi (in alcune Regioni)

COVID 2019

Snap4City (C), October 2020
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

![Diagram showing tracking of devices and mobiles]

Apps

OBU

IOT Device

MOBILE

Snap4City (C), October 2020
Tuscany in a Snap Mobile App on Android
Managing IOT Applications
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 Management

• Properties
  – Name, Type, Creation date

• Control
  – Restart
  – Delete

• Change of ownership
  – Toward another Snap4City User

• From inside the flow
  – Restart
  – 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 Management of IOT Edge

SNAP4CITY

IOT Edge Device

On premise, on your local plant/home

Remote Access to your IOT Application via secure connection

Local Access on IOT Edge Device on your local network

Intranet

Internet

Click on it

FireWall Gateway

On Cloud

Snap4City (C), October 2020
Advantages of IOT Edge remote control

- You do not need to be/move in the local network to access at the programming of Your IOT Edge Devices
- You can update the logic flow of your IOT Edge Devices from remote, so that you can perform remote maintenance of your network

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
HOW To install IOT Edge 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
5. Go in the list of Your IOT Applications on Snap4City.org or other cloud installations
6. Identify the IOT Edge IOT App and click on it to open the view on the IOT Applications flows
Creating IOT Applications
IOT Applications Development

MicroServices collections
My IOT Applications
IOT App. Editor
ServiceMap Discovery
Dashboard Collection, Editor and Wizard
Resource Manager
Generating IOT App With Dashboard
Sharing/saving reusing IOT App

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

+ on IOT Edge Raspberry

Kwm2m

Kwm2m client

Kwm2m client
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
DEMO
Section 1
Node-RED Hello World
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
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 1/2

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

debug
- Output: msg, payload
- to: debug tab
- Name: Name

switch
- Name: Name
- Property: msg, payload

random
- Generate: a whole number - integer
- From: 1
- To: 100
- Name: Name
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.
Resulting Dashboard

This is a local Node-RED dashboard.

The dashboards created within the Snap4city platform are more:

- Powerful
- Flexible
- Secure
end DEMO

Section 1
IOT App = Node-RED + Snap4City
In the IOT Application of Snap4City, it is possible to:
- Create multiple concurrent Flows for each IOT Application
- Execute flow that process data as: Event Driven, Batch (periodic or not)
- Load other libraries of MicroServices/Nodes/Blocks
  - The loading is allowed only for Administrators for security reasons
- Save/load, share, Flows, and applications with other users via the Resource Manager or with JS Foundation
- Ask a limited number of IOT Applications.
  - The Limit may depend on the organization or on personal authorization
Load an IOT application of example

aaa

- Import s4c
- Import S4C
- Manage palette
MicroServices Areas

Daisy of MicroServices

https://flows.nodered.org/?term=snap4city
Smart City and IOT main needs

**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.

https://flows.nodered.org/?term=snap4city
Basic Node.js Blocks on NodeRed on our Advanced IOT Apps

+ on IOT Edge Raspberry
Aug 2020 collection
Two Snap4City Libraries

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

We suggest also to install:

https://flows.nodered.org/search?term=snap4city
• ANY kind of sensors
• To Get DATA of a Service / POI /sensor
  – Historical and real time
  – Real Time
• 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, ...
Concepts of Services: Macro and subcategory

A SKOS area into the Km4Clty Ontology and Knowledge base for modeling POI and any element on map.
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*
• Distance from GPS point

• Point is in Polygon?
  – Polyline as WKT
Smart City Entities Search

Simple and Fast

- 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
S4CSearch

• 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
• **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
• 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
Controlling Maps from IOT Apps

- Show points on maps
- Get Points
- Tracks
- See examples on:

https://iot-app.snap4city.org/nodered/nrve0e3/ui/#1/0

https://www.snap4city.org/409
https://www.snap4city.org/417
Native Local       or          Snap4City

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

- Local on IOT Edge

- 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

- **Widgets:**
  - Impulse Button
  - Button
  - Switch
  - Dimer/Knob
  - KeyPad
  - geolocator

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

From Dashboard to IoT App

MyKPI variable

Synoptics

Nature

 IoT Application

Snap4City (C), October 2020
Dashboard-IOT App

Nature

From IOT App to Dashboard

IOT Application

gauge chart
single content
speedometer
horizontal
single bar
vertical single bar
web content
time trend
bar series
radar series
pie chart
curved line series
table content
synoptic write

Snap4City (C), October 2020
**Single Content Widget (flexibility)**

From Dashboard Editor and IOT Applications, accepts in input:

- Numbers
- String
- HTML code

https://www.snap4city.org/578
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

- 3) The click on the map can be used for passing GPS coordinates into IOT App to:
  - search for location
  - picking the value of one or more heatmaps
  - dynamically change data on widgets
  - Etc.
• 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
MicroServices Suite for Smart City Applications

- https://doi.org/10.3390/s19214798
- https://www.mdpi.com/14248220/19/21/4798/pdf
Dynamic Widgets data on Dashboard from IOT Applications
How the Dashboards exchange data

Snap4City BigData Storage and KB

- Req. ServiceURI
- Req. KPI, Metric ID
- Req. MyKPI ID
- GIS, HTTPs URLs

MyKPI, MyPOI, ...

API, External Services, MicroApp

- ServiceURI (ID)
- MyKPI, Metric (ID)
- Dynamic Data, computed into IOT Application
- Rx. Dynamic Data
- Event Driven Synoptics
- Actions, Show

IOT Application

Dashboards

Snap4City (C), October 2020
<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>MyPersonal Data</th>
<th>MyData</th>
<th>My KPI</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>![Icon]</td>
<td>Speed Limit</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>![Icon]</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>![Icon]</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>![Icon]</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>![Icon]</td>
<td>Single and Multiple Bars, stacked or not</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>![Icon]</td>
<td>MultiSeries, shaded, staked and non staked</td>
<td>X (cs)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>![Icon]</td>
<td>Time Trend (single)</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>![Icon]</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>![Icon]</td>
<td>SpiderNet, radar, Kiviat</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>![Icon]</td>
<td>Pie, Donut, 2 layers Donut</td>
<td>X (cs)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>![Icon]</td>
<td>Table</td>
<td>X (cs)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</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 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 (service URI), 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 realoading

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_ow_33665468’
  - metricHighLevelType: “sensor”
  - metricName: “tusc_weather_sensor_ow_33665468’
  - metricType: “airTemperature”

- ServiceURI (ID)
  - metricId: “https://servicemap.disit.org/webapi/efo/api/v2/serviceurl’
  - metricHighLevelType: “sensor”
  - metricName: “tusc_weather_sensor_ow_3362522’
  - metricType: “airTemperature”

- MyKPI (ID)
  - metricId: “7856230”
  - metricHighLevelType: “MyKPI”
  - metricName: “SmartTuscanyTrackerLocation”
  - metricType: “Velocity”

- Dynamic
  - metricId: “ ”
  - metricHighLevelType: “Dynamic”
  - metricName: “BatteryTemperatureGalaxyNote”
  - metricType: “Gradi Centigradi”
  - metricValueUnit: “°C”
  - measuredTime: “2010-11-21T14:51:00Z”
  - value: 6.08898131304505

- Dynamic
  - metricId: “ ”
  - metricHighLevelType: “Dynamic”
  - metricName: “BatteryTemperatureGalaxyNote”
  - metricType: “Gradi Centigradi”
  - metricValueUnit: “°C”
  - measuredTime: “2010-11-21T14:51:00Z”
  - value: 62.082079411516

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

Snap4City (C), October 2020
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
Sensors Actuators Allow to change the set up
IOT Application Integration with Synoptics
From-To Custom Widgets / Synoptics to Storage in WS

MyKPI

Sensors

MyKPI

Sensor

New Shared Variables

Constant Values
Case 1

Case 2

Case 3

2500 messages per second
Case 1

Case 1 SVG ws3


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


40 messages per second
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
Example of complex IOT Application

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

Snap4City (C), October 2020
Step 2

- Random
- Connect
- Configure
- Deploy
- Click
- Observe
Step 3

• Single content
• Connect
• Configure
• Deploy
• Click
• Observe

Snap4City (C), October 2020
Nodes configuration

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

- **Debug**
  - Output: msg, payload
  - to: debug tab
  - Name: Name

- **Random**
  - Generate: a whole number - integer
  - From: 1
  - To: 100
  - Name: Name

**Nodes configuration diagrams**

Snap4City (C), October 2020
Nodes connections

- SingleContent - Random Value
  - connected to ws://dashboard.km4city.org:8080/server
- GaugeChart - Random Value
  - connected to ws://dashboard.km4city.org:8080/server
- SpeedMeter - Random Value
  - connected to ws://dashboard.km4city.org:8080/server
- Time Trend - Random Value
  - connected to ws://dashboard.km4city.org:8080/server

(timestamp)
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

56
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

Copy the path
Copy the value
Step 1 Bis

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

- Get My KPIData
- Connect
- Deploy
- Click
- Observe
• Inject Node

• Configure with data of Weather Sensors and MyKPI retrieved at the previous steps
Step 4

- Bar Series
- Connect
- Configure
- Deploy
- Click
don't know
- Observe
Nodes connections

Node-RED
Resulting Dashboard
end DEMO

Section 2
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

Snap4City (C), October 2020
R Studio and Python algorithms are automatically transformed into MicroServices for your IOT Applications.
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
```
More Information

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

Automatic_IDW_Heatmaps_Creation.R

```r
#' @get /heatmapIDW
#' @serializer unboxedJSON
heatmapIDW <- function (city, long_min, long_max, lat_min, lat_max, epsgProjection, subnature, valueType, fromDateTime, toDateTime, heatmapName, colorMap) {
  ...
}
```
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
- **fromDateTime** = 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, Km4City

- Swagger
- SPARQL, FLINT
- Ontology Schema
- LOG.disit.org
- Smart City API from Knowledge Base and other tools
- Creating Micro Services
- Using them into IOT Applications
- Saving
- Sharing
- Reusing
- Big Data Store Facility
- Resource Manager
- Coding
- Testing

Snap4City (C), October 2020
• 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

My Scraping process

Web Scraper PORTIA

Generating WEB Scraping

Knowledge Base, Km4City

IOT App. Editor

Sharing/saving reusing Scraping

Resource Manager

Snap4City (C), October 2020
Web scraping

- TC9.16 – Web Scraping to get data from web pages
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.
• [https://www.mdpi.com/1424-8220/19/21/4798/pdf](https://www.mdpi.com/1424-8220/19/21/4798/pdf)
IOT App Examples
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.
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.
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.
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).
From Simple to Data processing IOT Applications
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

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

Copy the path

Copy the value
Step 3

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

```javascript
msg.payload = msg.payload.realtime.results.bindings[0].freeParkingLots.value
```

Snap4City (C), October 2020
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
Nodes configuration

- **Payload**: `timestamp`
- **Topic**: Blank
- **Repeat**: `interval` every 15 minutes
- **Name**: Blank
- **Service Uri**: `http://www.disit.org/km4city/resource/CarParkPieracciniMeyer`
- **Language**: Italian
- **Function**: `Get Free Parking Lots`

```
1 2   bindings[8].freeParkingLots.value
3  return msg;
```
Nodes connections
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.
Resulting Dashboard

BasicDemo23Luglio

Fri 19 Jul 11:00:36

https://main.snap4city.org/view/index.php?iddashboard=MTk1OQ==
end DEMO

Section 3
Production of IOT Applications
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

Snap4City (C), October 2020
Nodes configuration

Free Park

Name: Free Park

Function:
```javascript
msg.payload = 
2 "<b style='color: green' >Free" + msg.payload + "</b>"
3 return msg;
```

Busy Park

Name: Busy Park

Function:
```javascript
msg.payload = 
2 "<b style='color: red' >Full" + msg.payload + "</b>"
3 return msg;
```
Resulting Dashboard

Threshold

Free 207

Start DEMO
Section 4
Start DEMO
Section 4
Data Processing with IOT Application (DEMO)
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

- **Payload**: timestamp
- **Topic**: interval
- **Repeat**: every 15 minutes
- **Invert once at start?**
- **Array**: Split using Fixed length of 1

**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;
```

**Dashboard**: Name: TotalFreePark

**Widget**: Gauge - TotalFreePark

**Service Info**:
- **Name**
- **ServiceUrl**: http://
- **Language**
Nodes configuration 2/2
Nodes connections

timestamp → service-search-within-circle

split → service-info

switch → join → Sum Of Free Park

Gauge - TotalFreePark
connected to ws://dashboard.km4city.org:8080/server

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

Speedometer - TotalFreePark
connected to ws://dashboard.km4city.org:8080/server

TimeTrend - TotalFreePark
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.
Result

TotalFreePark

Fri 19 Jul 16:03:24

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
One Possible Solution
Nodes configuration 1/2
Nodes configuration 2/2
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 with Ticketing Systems Workflows
Integration with Ticketing Systems Workflow

- You ticketing systems can be integrate with Snap4City, by means of IOT Applications and Dashboards
- [https://www.snap4city.org/597](https://www.snap4city.org/597)
OpenMain Ticketing System

- Activate events
- Manage steps
- Collecting feedbacks and results from teams
Example of Integrated workflow

- Consumptions/productions
- Event detection and firing
- Critical event management
- Workflow management, team assignment, material control
- Business Intelligence
- Predictive Maintenance

Snap4City (C), October 2020
Integration with Twitter Vigilance
IOT and data World

My IOT Devices

IOT Applications

Big Data Analytics, Artificial Intelligence

Dashboards and Apps

Twitter Vigilance
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
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 Multiseries 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
Advanced Snap4City APIs and MicroServices

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

Snap4City Portal and Integrated tools

Datagate

Open or Private External CKAN Data Portals

IoT App

IoT App

Knowledge Base, Km4City

Knowledge and Storage Data from the Field and City

Real Time Data

Heatmaps

What-IF Analysis

Remote Control

Historical Data

Snap4City Dashboards

Mob & Web Apps
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
General solution, bring data from API to Dashboards

• 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 be automatically transformed into a MicroService for the IOT Applications
Edit MicroService Call: call and help editing

Formal definition

HELP
Usage of the MicroService from IOT App

Snap4City (C), October 2020
IOT Network Support
IOT Basic

• 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

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 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
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
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
- IoT Directory
- Service Map
- Resource Manager
- Data Gate
- ETL

LIVING LAB - DEV TOOLS - COWORKING
- IoT Directory
- Service Map
- Resource Manager
- Data Gate
- ETL

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

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

- GIS
- City Utilities
- Open Data
- Legacy & External Services
- Personal Data
- IOT / IOE
- Brokers
- KPI
- Industry 4.0
- Social Media
Snap4City: Builder of Sentient Cities Solutions

Dashboards with data driven IOT Applications enforcing intelligence

IOT and data World

My IOT Devices

IOT Applications

Big Data Analytics, Artificial Intelligence

Dashboards and Apps

Snap4City (C), October 2020
Snap4City: IOT Directory and data/device Discovery

(IOT) Discovery
IOT Directory
Knowledge Base
Semantic Reasoners

IOT Applications
Dashboards and Apps

IOT Networks
IOT Brokers
IOT Gateways
Snap4City: IOT Directory and data/device Discovery

IOT Devices
IOT Edge Devices
Big Data Analytics, Artificial Intelligence

(SNAP4City (C), October 2020)
Standards and Interoperability


https://www.snap4city.org/65
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 Broker

Registering

Discovering

Knowledge Base, Km4City

Knowledge and Storage Data from the Field and City

IOT Directory

Discovering

My IOT Device

Register

IOT Network Manager

Final user Manager

IOT Application

Discovering

Dashboard Wizard

Register

IOT Directory

Discovering

ServiceMap

Knowledge Base

Discovering

Browsing

External

Internal
Main Features of the IOT Directory

• Registers IOT Brokers
  – Different kind of Brokers, different kinds of authentications and protocols
  – Registered IOT Orion Brokers can be queried for collecting their managed devices, so that those IOT Devices are registered

• Registers IOT Devices: singularly or at groups (in Bulk)
  – Registration can be custom or based on IOT Device Model
  – IOT Edge are registered as special IOT Devices
  – Registered IOT Devices are saved into local Data base and Knowledge Base

• Provides support for security aspects:
  – Generation of Certificates, Keys, etc.
  – Collection of keys when IOT devices are on some IOT Gateway or Second Level IOT Broker.

• Manages Ownership and Delegation for
  – IOT brokers, IOT devices, IOT Device Values
## IOT Directory Features vs Users Roles

<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>Discovery</td>
<td>KB, API, ..</td>
<td>X</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></td>
</tr>
<tr>
<td>Register in Bulk</td>
<td>API, ..</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delegate, Change Owner</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>Yes</td>
</tr>
<tr>
<td>Discovery</td>
<td>KB, API, ..</td>
<td>X</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>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create, change, delete</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>delegate, change ownership</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOT Broker</td>
<td>Browse, use</td>
<td>use</td>
<td>Browse, use</td>
<td>X</td>
<td></td>
<td>Yes (use)</td>
</tr>
<tr>
<td>Register/change/Delete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delegate</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Periodic Update</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

**Snap4City (C), October 2020**
In which case you are?

Case B2

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

IOT Device

IOT Broker

Case B1

IOT Device

IOT Broker

i) Registered IOT Broker on Snap4City

Case A1.2

IOT Device

IOT Broker

a) Registered IOT Device on Broker

Case A2

IOT Device

IOT Broker

a) Registered IOT Device on Broker

i) Registered IOT Broker on Snap

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 a 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
   3. 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.

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).
Data Ingestion Strategy
Snap4City Architecture vs Data Ingestion

Data Sources, External Services
- PULL Data

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

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

Knowledge base
- Semantic reasoners

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

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

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

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

Back-End
- Elastic search

Semantic reasoners

Facet, semantic search
Snap4city Data Ingestion Flow Diagram

Static Models

- Static or quasi Static Data: POI, etc.
- IOT Broker Registration
- IOT Device Model Reg.
- IOT Device Registration

IOT Directory

Knowledge Base

Semantic Reasoners

Indexing and Aggregating

NIFI, Elastic Search

Real Time data in Pull

Any gateway, server

periodic

Real Time

Time Series

IOT Orion Broker

Information, File

IOT Apps

Snap4City Tools

IOT Device/Gateway

IOT Device

Adapter

Automatic

Subscription note

SURI Link

IOT Orion Broker

IOT Orion Broker

IOT App

Adapter

Adapter

IOT Device

IOT Device

IOT Brokers

Information, File

IOT Apps

Snap4City Tools

IOT Device/Gateway

Snap4City (C), October 2020
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
Checking data ingestion results

- Data Inspector
- ServiceMap, ASCAPI
  - LOG / LOD viewer
  - Super Service Map
- IOT Broker
- ACAPI: Swagger
- IOT Directory
- Data Inspector
- ServiceMap, ASCAPI
- DevDash (Kibana)
- Elastic Search

Some functionalities are limited to certain roles
• The IOT Orion Broker 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

• IOT App, IOT Agents, Adapters and IOT Brokers can
  – be on IOT Edge
  – be on other clouds and services
  – work on a large range of different protocols and kinds
  – have or not Snap4City tools and 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/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

• **Data Storage:** 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
IOT Broker Registration
<table>
<thead>
<tr>
<th>IOT Broker</th>
<th>Access Link</th>
<th>Access Port</th>
<th>Protocol</th>
<th>Ownership</th>
<th>Organization</th>
<th>Owner</th>
<th>Created</th>
<th>Edit</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>iotobsf-smartbed</td>
<td>192.168.1.47</td>
<td></td>
<td></td>
<td>DELEGATED</td>
<td>SmartBed</td>
<td>angelo.difino</td>
<td>2019-11-29</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>mqttUNIFI</td>
<td>192.168.110</td>
<td></td>
<td></td>
<td>DELEGATED</td>
<td>DISIT</td>
<td>iotdirectory.dist</td>
<td>2018-02-07</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>mqttUNIMI</td>
<td>159.149.129.184</td>
<td></td>
<td></td>
<td>DELEGATED</td>
<td>DISIT</td>
<td>iotdirectory.dist</td>
<td>2018-04-30</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>orionAntwerp-UNIFI</td>
<td>broker3.snap4city.org</td>
<td></td>
<td></td>
<td>PUBLIC</td>
<td>Antwerp</td>
<td>iotdirectory.antwerp</td>
<td>2019-06-03</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>orionFinland</td>
<td><a href="https://ngsi.fh.fi">https://ngsi.fh.fi</a></td>
<td></td>
<td></td>
<td>PUBLIC</td>
<td>Helsinki</td>
<td>iotdirectory.helsinki</td>
<td>2018-11-21</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>orionFirenze-UNIFI</td>
<td>192.168.117</td>
<td></td>
<td></td>
<td>PRIVATE</td>
<td>Firenze</td>
<td>iotdirectory.firenze</td>
<td>2019-10-28</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>orionHelsinki-UNIFI</td>
<td>broker2.snap4city.org</td>
<td></td>
<td></td>
<td>PUBLIC</td>
<td>Helsinki</td>
<td>iotdirectory.helsinki</td>
<td>2019-06-03</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
<tr>
<td>orionLonatoDelCarda-UNIFI</td>
<td>192.168.146</td>
<td></td>
<td></td>
<td>DELEGATED</td>
<td>LonatoDelCarda</td>
<td>iotdirectory.lgc</td>
<td>2019-10-28</td>
<td>EDIT</td>
<td>DELETE</td>
</tr>
</tbody>
</table>

IP: 192.168.112
Latitude: 43.76666
Longitude: 11.26242
Login: SHA
Password:
Register IOT Broker

Snap4City (C), October 2020
Please note on IOT Brokers

• An IOT Broker
  – is 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 an IOT Orion Broker which
    • is compliant with a specific NGSI version: V1, V1-2, V2, etc...
  – can be an IOT Orion Broker configure to
    • expose different authentication methods: K1/K2, Certificate, etc.
    • be accessible from IOT Devices and IOT App in Cloud only
    • be accessible from Internet
IOT Orion Broker Network

• IOT Broker can be Internal (on Snap4City Cloud)
  – Registered and automatically connected on NIFI and Data Shadow
  – Registration of IOT Devices is passing from IOT Directory

• 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

• IOT Brokers can be networked
  – Services, Service paths: for managing the IOT Broker network (in progress)
  – Multi-tenacy: more than one user/org on the same IOT Broker (to be released)
**NGSI versions**

- Orion Broker of V2 with NGSI syntax of V1
- Orion Broker of V1 with NGSI syntax of V1 + Secure Filter of Snap4city
- Orion Broker of V2 with NGSI syntax of V2
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)

Each new measure in Snap4City is conventionally time located in «dateObserved»

<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)
# IOT Device Data 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: OrionUNIFI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOT Broker</td>
<td>Protocol: NGSI</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>
**Model meaning**

- **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 time stamp of the time serie is conventionally «dateObserved» in Snap4City
  - In new *SensorMobile* HLT, also GPS can be changing over time as in the MyKPI
IOT Device Registration
Activities for IOT Device Registration

• Registration of
  – 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.
IOT Discovery on IOT Application Development
IOT Device Model Registration
Many IoT Devices?  IOT Device Model!!!

Example: ChargingStationModel

- **Prerequisites**: 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
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 from model
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

Available example: 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)
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)
IoTApp Dynamic Flow

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

5) Working on Dynamic Flow to save Average #people every 15 minutes for each IoTDevice
A Complete Example for Time Series: IOT Device Model + IOT Data Ingestion
I have created an IOT Device Model as:

Statuscorregione
- Name: statuscorregione
- Description: Ok
- Sensor: misura
- Kind: Device Type
- Frequency: protezione civile
- Healthiness Value: Producer
- Automatic Generation: Healthiness Criteria
- Edge-Gateway Type: Key Generation

Edit Model - statuscorregione

Snap4City (C), October 2020
For Time Series
- **ValueName**: dateObserved
- **DataType**: time
- **ValueType**: timestamp
- **ValueUnit**: timestamp in millisecond
- E.g.: ISO string of the date-time
From IOT Model I have created some instances: the IOT Devices
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 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» 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.
- Symbolic address of IOT Broker can be taken from IOT Directory.
• Orion Broker of V2 with NGSI syntax of V1

• Orion Broker of V1 with NGSI syntax of V1 + Secure Filter of Snap4city

• Orion Broker of V2 with NGSI syntax of V2
A Json from the IOT App

**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": "time" },
        { "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": 22222344,  "type": "integer" }
    ]
}
return msg;
```
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

```
    msg.auth = {
        "k1": "1ef0e5e8-yyyy-xxxx-9462-0aa4cf5e19",
        "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
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
Comparison With other platforms
<table>
<thead>
<tr>
<th></th>
<th>OT Discovery Abstraction</th>
<th>IOT Device Management</th>
<th>IOT and Open SW</th>
<th>Integrated Community Management</th>
<th>Data Types: IOT Devices, IoT and Open SW</th>
<th>Data Types: Publish/subscribe, Data: Open, OSGi, Delegation, Consent and change</th>
<th>Auditing on Data Type Access</th>
<th>Scalability IOT</th>
<th>Usual Programming end-to-end</th>
<th>Multi-Domain Semantic Platform</th>
<th>Standard based Modules and APIs, End Devices</th>
<th>Open Source IOT</th>
<th>Multi-protocol on IOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snap4City</td>
<td>G</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Thingsboard  [55]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>IOT eclipse.org [56]</td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Limited</td>
</tr>
<tr>
<td>IoT IGNITE [57]</td>
<td></td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Limited</td>
</tr>
<tr>
<td>FIWARE [47]</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Limited</td>
</tr>
<tr>
<td>ARM mbed IoT [48]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>(N)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>(N)</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Airvantage [51]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Limited</td>
</tr>
<tr>
<td>AWS [43]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Azure IOT [44]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>PTC ThingWorkx [59]</td>
<td></td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Bosch IoT Suite [58]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>(Y)</td>
<td>(N)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>CISCO Jasper [55]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Limited</td>
</tr>
<tr>
<td>Siemens MindSphere [60]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>(Y)</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Carriots [54]</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>(Y)</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Google IOT [45]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Homekit Apple [50]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Limited</td>
</tr>
<tr>
<td>Smarthing Samsung [52]</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>(Y)</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Limited</td>
</tr>
</tbody>
</table>
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, protocols and data formats

- 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
Fi-Ware vs Snap4City
SMART CITIES REFERENCE ARCHITECTURE

• Is open to the Development of Applications leaving large space to developers
• Is centered on the Orion Broker that result central in the architecture: any Broker or data source is sending data to Orion
• No data shadow at the beginning, only recently they are adding data shadow on IoT Broker
• Security level is not clear, partially demanded to developers
• Visual Flexible IoT processing is not clearly provided
• Limited API for IoT data access
• Knowage BI presents several limitations in showing Smart City Data
• Market place on Open Data
• Support of Developers via Fi-Ware
• Deployed as VM and Dockers
• open source, not the application parts

---

VS

• Is open to the Development of Applications leaving large space to developers
• Is centered on the Orion Broker that result central in the architecture: any Broker or data source is sending data to Orion
• No data shadow at the beginning, only recently they are adding data shadow on IOT Broker
• Security level is not clear, partially demanded to developers
• Visual Flexible IoT processing is not clearly provided
• Limited API for IoT data access
• Knowage BI presents several limitations in showing Smart City Data
• Market place on Open Data
• Support of Developers via Fi-Ware
• Deployed as VM and Dockers
• open source, not the application parts

• Is open to the Development of Applications leaving large space to developers
• Is centered on the Orion Broker that result central in the architecture: any Broker or data source is sending data to Orion
• No data shadow at the beginning, only recently they are adding data shadow on IOT Broker
• Security level is not clear, partially demanded to developers
• Visual Flexible IoT processing is not clearly provided
• Limited API for IoT data access
• Knowage BI presents several limitations in showing Smart City Data
• Market place on Open Data
• Support of Developers via Fi-Ware
• Deployed as VM and Dockers
• open source, not the application parts

• Is fully distributed, any kind of data source can be ingested, automatically to form the Data Shadow.
• Orion Broker is only one of the Brokers that can be used. It can be also protected by Snap4City tech, with Mutual Authentication
• Visual Flexible IOT processing is provided as IOT App that is Node-RED plus Snap4City MicroServices suites
• Advanced Smart City API are provided on top of Knowledge Base
• Dashboard Builder has been designed for Smart City Data and automated dashboards’ production
• Market Place for promoting, publishing and sharing Open Data, tools, processes, experiences
• Full Support for Living Lab of the city, coworking, tutorials
• Deployed as VM and Dockers, on cloud and on premise
• Fully support for Multi-tenancy
• 100% open Source

Snap4City (C), October 2020
• **Snap4City is an official Fi-Ware Solution via**
  – NGSI V1, V2 The IOT Orion Broker
  – IOT Orion Broker can connect JSON, MQTT, Lightweight M2M, LoraWAN, OPC, SigFOX, etc. see Fi-Ware [Https://www.fiware.org](https://www.fiware.org)

• **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)
• 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.
Proprietary IOT Devices as well as Open Hardware / Open Software
IOT Devices

- LoraWAN + Arduino + I2C, NGSI
- Snap4All IOT Button ESP, NGSI, Wi-Fi, BT
- Any Sensor / Actuator Open to other protocols

IOT Edge Devices

- IOT Edge NodeRED: Raspberry Pi, ZigBee, WiFi, RJ45,.., Android, LINUX, Windows, ...

Snap4All PAX Counter LoraWAN WIFI, NGSI, GPS

LoraWan Gateway: IOT Edge, NGSI, Wi-Fi, RJ45, GPS
IOT Dev Management: activities

- IOT Devices 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 monitored automatically
- IOT Devices can be public or private
  - Full support of Proprietary protocols and devices
  - Providing Open Hardware and Open Software IOT Devices/IOT Edge: NGSI fully secure
- 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
Lora IOT Device, Arduino

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

https://www.snap4city.org/216
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

https://www.snap4city.org/276
https://www.snap4city.org/297 help config
CNR IBE AirQuino
CNR developed a circuit board "AirQuino", 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, CO\textsubscript{2}, O\textsubscript{3}, NO\textsubscript{2}, CH\textsubscript{4},
– 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/508  
https://www.snap4city.org/download/video/tn/ARQuino-CNR.pdf
Libelium Sensors
Libelium

- PM10
- Temp
- Humidity
- Pm2.5
- NO
- NO2
- CO2
- Etc.
IOT Gateway / IOT Edge
LoraWan Gateway out of the Box

- Raspberry Pi Based LoraWan Gateway
- Physical UpLink as: Wi-Fi, RJ45
- Logical UpLink: LoraWAN TheThingsNetwork, NGSI V2 (mutual authenticated Snap4City) toward IOT Broker
- Powered 5V
- GeoLocated GPS Antenna
- IOT Edge Snap4City Included if needed
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

Alexa: Voice Commands

Local Control

DCS

Administration Servers

ODBC

WiFi

Modbus

SCADA

OPC UA

Controlling
Energy Power

Measuring
Energy Consumption

Snap4City (C), October 2020
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....
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

- 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
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
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

• Fixed PaxCounter LoraWan
  – Based on Wi-Fi- Bluetooth

• Mobile PaxCounter LoraWan
  • Based on Wi-Fi- Bluetooth

• Fixed PaxCounter (LoraWan+Wifi out)
  – Based 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 Active From 2019-09-23T00:00:00+02:00Z and 2019-09-23T05:30:00+02:00Z

Device in Cumulative Mode OFF
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
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
IOT end-2-end Secure Stack
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
  - IoT Edge
  - Aggregator and distributor: On premise computing

- Data Shadow
  - IoT Firewall, IoT Context Broker
  - IoT App
  - storage
  - Via MicroServices: Cloud computing

- User Interface
  - Dashboard Builder
  - HTTPS
  - WSs
Making IOT data more available
Executing local computation

WSs, HTTPS
WSs, HTTPS
WSs, HTTPS

IOT Devices
[IoT Edge]
IOT Broker
[IoT Application]
Dashboard Engine

WSs, HTTPS
WSs, HTTPS
WSs, HTTPS

Intranet
Internet
Cloud

Smart City Knowledge Base and RT data

Shadow

MicroServices Ext-Services
Executing permanent computation

Ext Services

User interface On Browser

WSs HTTPS

Sensors/Actuators
Smart City IoT Platform Respecting GDPR Privacy and Security Aspects

CLAUDIO BADIO, PIERFRANCESCO BELLINI, ANGELO DIFINO, AND PAOLO NESI

Department of Information Engineering, University of Pisa, Pisa, Italy
Corresponding author: Paolo Nesi (paolo.nesi@unipi.it)

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Agreement 888996.

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 claim 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 Dashboards, 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 verifying 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/5G/IoE (Internet of Things/Everything) in the hands of public administrations and Living Labs, and selected a number of solutions. Consequently, at the end of the process 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 pervasiveness of IoT Devices, integrated in common objects, is becoming increasingly deeper. The addresses’ space for these devices would be enough to point any amount of any devices at any moment without restrictions. Dense products that implement Low-Power Wide Area Networks (LPWAN) technologies for IoT introduced by SigFox and Semtech (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, [23] 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.
Acknowledgements
DISIT lab roadmap vs model and tools’ usage

2013
- Km4City Ontology 1.1
  - Tuscany, Road Graph
  - Mobility
  - culture, tourism
  - Events
  - Parking
  - Services
  - Linked open graph

2014
- Weather Forecast
- Real Time Wi-Fi
- Entertainment
- LOD
- Twitter Vigilance
- Social Media Analytics, Sentiment Analysis

2015
- Resilience
decision Support
- Smart First Aid
- User Behaviour Analysis, predictions
- Risk Analysis

2016
- SII-MOBILITY SCN
  - Infomobility
  - Mobile App
  - Routing
  - Multimodality

2017
- GREEN IMPACT
  - POR FESR 2014-2020
  - Industry 4.0
  - Critical Plant
  - Monitoring

2018
- Mobility Demand / Offer
  - Mobility
  - Strategy
- Origin-Destination and trajectories
- Traffic Reconstruction
- Offer Analysis
- OBU, smart devices

2019
- IOT/IOE
  - Sardinia Region
  - Smart City Strategies and plan
- GHOST SIR
  - SII-Mobility

2020
- H2020
  - Mobility
  - Strategy

2021
- CAPELON
  - Sweden
- Smart Mobility
- PISA, PUMS
- Living lab

2022
- Smart Lonato
- PCP Award
- Smart Tourism
- 6 Pilots
- Data Analytics
- Extended platform

...2022
Main running projects

- 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
- 5G → Industry 4.0 vs SmartCity
- Green Impact → Industry 4.0, Chemical Plant
- 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
Acknowledgements

- Thanks to the European Commission for founding. All slides reporting logo of Snap4City [https://www.snap4city.org](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/](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.
DISIT thanks to

Herit Data: Tourism and Mng.  [https://herit-data.interreg-med.eu/](https://herit-data.interreg-med.eu/)
Snap4City: IOT/IOE smart city [www.snap4city.org](http://www.snap4city.org)
Trafair: CEF project with several Cities [http://trafair.eu/](http://trafair.eu/)
Mosaic: Mobility and transport model
Km4City: [http://www.km4city.org](http://www.km4city.org)
REPLICATE H2020, SCC1, EC flagship  [http://replicate-project.eu/](http://replicate-project.eu/)
Sii-Mobility SCN MIUR: [http://www.sii-mobility.org](http://www.sii-mobility.org)
Feedback: retail and GDO Big Data analytics
5G with 3G-Wind, Open Fiber, Estra
Coll@bora Social Innovation, MIUR:  [http://www.disit.org/5479](http://www.disit.org/5479)
TRACE-IT, RAISSS, TESYSRAIL, ...
Mobile Emergency:  [http://www.disit.org/5404](http://www.disit.org/5404)
Further readings

- 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
Be smart in a SNAP!

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

www.snap4city.org

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